

NASA Technical Paper 1219

A Digital Computer Program for
the Dynamic Interaction Simulation
of Controls and Structure (DISCOS)

Volume II

CASE FILE
COPY

Carl S. Bodley, A. Darrell Devers,
A. Colton Park, and Harold P. Frisch

MAY 1978

NASA

NASA Technical Paper 1219

A Digital Computer Program for the Dynamic Interaction Simulation of Controls and Structure (DISCOS)

Volume II

Carl S. Bodley, A. Darrell Devers,
and A. Colton Park
Martin Marietta Corporation
Denver, Colorado

Harold P. Frisch
Goddard Space Flight Center
Greenbelt, Maryland



National Aeronautics
and Space Administration

**Scientific and Technical
Information Office**

1978

All measurement values are expressed in the International System of Units (SI) in accordance with NASA Policy Directive 2220.4, paragraph 4.

PREFACE

This document, prepared by the Dynamics and Loads Section, Martin Marietta Corporation, Denver Division, under Contract NAS5-11996, presents the results of a study for the purpose of developing a computer program system for dynamic simulation and stability analysis of passive and actively controlled spacecraft. The study was performed from May 1973 to April 1975 and was administered by the Goddard Space Flight Center, National Aeronautics and Space Administration, Greenbelt, Maryland, under the direction of Joseph P. Young.

Upon delivery, the computer program and associated documentation was checked in detail by Harold P. Frisch. This document incorporates both the original Martin work and the supplementary material prepared at the Goddard Space Flight Center.

The digital computer program, DISCOS (Dynamic Interaction Simulation of Controls and Structure), has been extensively annotated and tested on a range of problems that should have exposed nearly all theoretical errors and programming bugs.

From its inception in 1973, DISCOS has been designed to grow as new needs and more efficient computational techniques developed. This feature makes it impossible to define a final version. To circumvent this problem the official release version will contain only those additions to the delivered program that enhance program documentation and user interface capability and correct programming errors.

Included in this version are more than 10,000 comment cards and a capability to routinely direct the computer to output on the line printer virtually all computation along with explanatory alphanumeric statements. A large percentage of the comment cards are in subroutines DEF1, DEF2, . . . , and DEF5. These subroutines are composed entirely of comment cards and provide the user with an area in the source file for keeping documentation current. In particular, subroutine DEF5 contains a narrative description of the program and its current capabilities.

For the uninitiated reader, it probably would be best to speed-read subroutine DEF5 to obtain a quick overview of the capabilities of the program and the solution techniques applied before reading this document.

The potential user should be aware of the fact that DISCOS is not intended for simple problems. It is primarily for problems which were heretofore intractable. Consequently, the theoretical basis for the program is highly advanced, and the computation algorithms are designed for the efficient processing of the equations associated with large multidegree-of-freedom systems.

As an aid to the user, the paper on which the derivation of the coupled flexible-body equations of motion is based, a paper that outlines the solution method, and comments on results obtained from several DISCOS applications appear as reference material at the end of Volume I. Volume I contains all relevant theoretical work.

Volume 2 describes the DISCOS program and its support programs. The user is encouraged to refer to the comment cards provided in all DISCOS subroutines for additional descriptive information. The comment cards found in the DISCOS subroutines are intended to provide a link between the computer code and the theoretical equations provided in Volume 1.

To effectively interface with the program the user must be able to write the subroutines that will define all non-gyroscope forces and torques. The user is provided with a clean interface. When the load vector associated with the effects of springs, dampers, motors, gas jets, etc. is defined and properly stored in the computer memory, DISCOS will automatically transform it into the appropriate generalized form required by the formulation.

The inclusion of effects such as aerodynamic loading and thermodynamic deformation is more difficult. However, the methodology is analogous to that used for including gravity gradient effects.

The methodology for including loads associated with springs, dampers, motors, gas jets, constraints, etc. is found in Volume II and in the comment cards of the appropriate subroutine referred to in Volume II.

DISCOS is probably the most powerful computational tool to date for the computer simulation of actively controlled coupled multiflexible-body systems. It is not an easy program to understand and effectively apply, but it is not intended for simple problems. The user is expected to have an extensive working knowledge of rigid- and flexible-body dynamics, finite-element techniques, numerical methods, and frequency-domain analysis.

CONTENTS

	<i>Page</i>
PREFACE	iii
I. PROGRAM SYSTEM OVERVIEW	1
A. Introduction	2
B. Simulation Overview and Nomenclature	3
C. General Use Information	8
II. PROGRAM SEGMENTATION	9
III. DELINEATION OF USER-SUPPLIED MODULES	10
A. Logic Flow	11
B. User-Supplied Module Specifics	12
C. Selected Common-Block Information	17
IV. PROGRAM INPUTS	19
A. Subroutines READ and READIM	19
B. Input Data Stream	20
V. PROGRAM OUTPUTS	20
VI. AUXILIARY PROGRAMS	33
A. REDIM—The Redimension Program	33
B. Support Programs for the DISCOS Flexible-Body Capability	
APPENDIX A—INPUT/OUTPUT SUBROUTINE EXPLANATIONS	A-1
APPENDIX B—SUPPORT PROGRAMS FOR THE DISCOS FLEXIBLE-BODY CAPABILITY	B-1

**A DIGITAL COMPUTER PROGRAM
FOR THE DYNAMIC INTERACTION SIMULATION OF
CONTROLS AND STRUCTURE (DISCOS)
VOLUME II**

Carl S. Bodley, A. Darrell Devers, and A. Colton Park
Martin Marietta Corporation
Denver, Colorado

and

Harold P. Frisch
Goddard Space Flight Center
Greenbelt, Maryland

I. PROGRAM SYSTEM OVERVIEW

This volume is intended to provide sufficient understanding of program system DISCOS (Dynamic Interaction Simulation of Controls and Structure) and its capabilities to permit a user to use the program as a basic tool for analyzing the behavior of a wide range of dynamical problems. Specific emphasis is on a simulation for multiple-interconnected spinning elastic bodies responding under the combined influences of external environments and either active or passive control.

The authors at Martin Marietta Corporation acknowledge the assistance provided by Goddard Space Flight Center personnel. Harold Frisch and James Donohue contributed many valuable technical comments and suggestions throughout the program. In particular, they developed the basic approach to be used for data input, defined exactly what should be included in the transfer function and stability analysis portion of the program, and defined eight of the eleven demonstration problems. They provided the associated data that were used to verify both the nonlinear time response and linear transfer function and the stability analysis portions of the program. Raymond Welch provided the subroutine used to generate root-locus plots. Dr. William Case provided valuable advice on interfacing with NASTRAN output and generated the demonstration problem used to validate the interface subroutine (NASFOR).^{*} During the early program development stage, Dr. James Mason offered significant advice on the need to compute internal forces at the interconnect points. Reginald

^{*}The NASFOR subroutine has been superseded by a special NASTRAN DMAP program and associated preprocessor program written by Harold P. Frisch.

Mitchell contributed invaluable advice on requirements for making the program compatible with the GSFC IBM 360/95 computer system. In addition, he supplied the contractor with a 360-compatible plot package, furnished the contractor with a self-authored subroutine for reading NASTRAN output, and was responsible for running all demonstration problems on the 360/95 computer. Finally, the authors acknowledge the encouragement and efforts of Joseph P. Young, Technical Monitor, who made numerous valuable comments and suggestions throughout the study.

A. Introduction

The simulation employs a state-space approach that was developed in detail in Volume I. The state-space formulation provides an attractive basis for simulating nonlinear dynamical problems in a general sense, as well as permitting linearization of the governing equations to provide an additional foundation with which to evaluate frequency-domain and linearized time-domain characteristics.

An attempt has been made to relieve the user of having to communicate with the digital program by means of large amounts of bulk data input. Although many options are available, the program data stream has been organized to require only a minimal amount of basic input data for a particular simulation. The data requirements have been further consolidated in a manner that is quite definitive for the physical system being simulated. In summary, the user can easily relate to the particular elements of the program requirements and, thus, minimize the setup time required for preparing data input for a given problem. In addition, a set of self-checking features has been included for identifying and checking certain compatibilities that are necessary for a proper simulation of a physically realizable system.

In an overall sense, the user can use the digital program to obtain:

- Nonlinear time response
- Interaction constraint forces
- Total system resonance properties
- Frequency domain response and stability information
- Linearized time response

The program outputs consist of printed and plotted* results depicting:

- Dynamic model construction
- Time-domain response
- Frequency-domain characteristics

*Plotting uses SC 4020 plot routines that are resident in the IBM-360 system at the Goddard Space Flight Center.

The printed outputs are of a fixed form, and the user controls the plotted information through the input data stream.*

B. Simulation Overview and Nomenclature

This discussion identifies the basic nomenclature used for synthesizing a typical assembly of interconnected bodies. The theoretical development and program annotation make extensive reference to various terminologies that are clarified here. Figure 1 provides a visual display that illustrates many of the items being discussed and will be referred repeatedly to in the ensuing discussions.

The user identifies the overall system “topology” by means of the input integer array, ITOPOL, that contains the necessary information describing which “hinges” interface which bodies. Each body contains a body reference point that is the origin of an orthogonal Cartesian body axis system. This point need not coincide with the body center of mass.

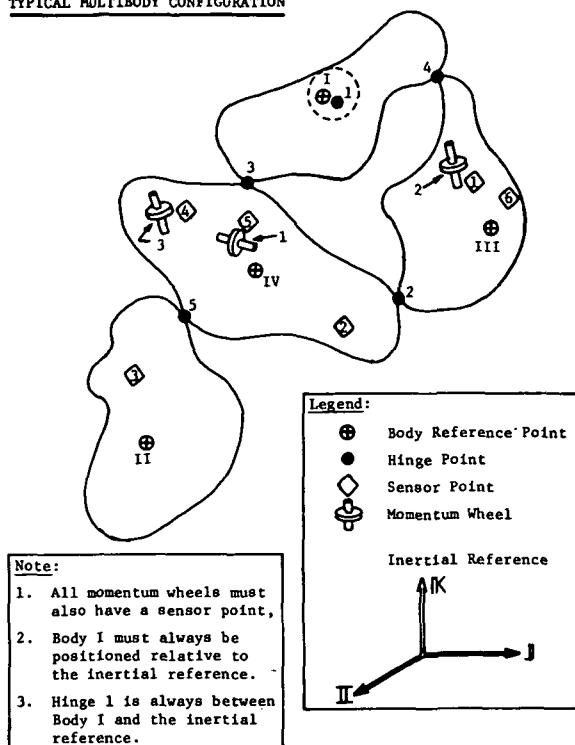
Contiguous “bodies” are interfaced through a “hinge.” “Interfaced” is used in lieu of “connected” to emphasize that the common “hinge” point between contiguous bodies may actually permit relative translational motion of the two bodies at the hinge. The user identifies the degree of fixity at the hinge by means of the input integer array, IHDATA. A typical body may contain “sensor” points that identify particular points at which additional information is required for completing the desired simulation. Although a sensor point might sense on position or rate for the control-system inputs, it could also represent a point on a body at which certain other information is desired, such as a momentum-wheel location or a point-of-force/torque application. The integer input array, IFTSMW, identifies the body on which a particular sensor point is located.

A body may also contain “momentum wheels.” This special consideration accommodates a disk or rotating mass with a single relative rotational degree of freedom into the simulation without introducing another body. The momentum-wheel capability is more efficient for simulating a single degree-of-freedom rotating mass than for constraining five of six rigid-body degrees of freedom by constraint equations. All momentum wheels must have an associated sensor point. A wheel may be either active or constant speed. An active wheel has a variable spin rate and receives an input torque (usually by some sensor output relationship). A shaft torque is applied to the wheel inducing an angular wheel acceleration. The array, IMO, identifies whether or not the wheel is active, and which axis is the spin axis. The reference axis for the wheel is the same as the sensor-point axis system on which the wheel is located. The array, AMO, identifies the wheel spin rates (initial rate only for the active wheel) and the wheel spin inertia about the spin axis.

The system state vector is arranged in a specific manner within the program, and the user must be very familiar with this arrangement for a number of reasons. First, the user must

*An option exists for printing most of the significant computations in the primary subroutines if desired.

TYPICAL MULTIBODY CONFIGURATION



TYPICAL TWO BODY/HINGE POINT ARRANGEMENT

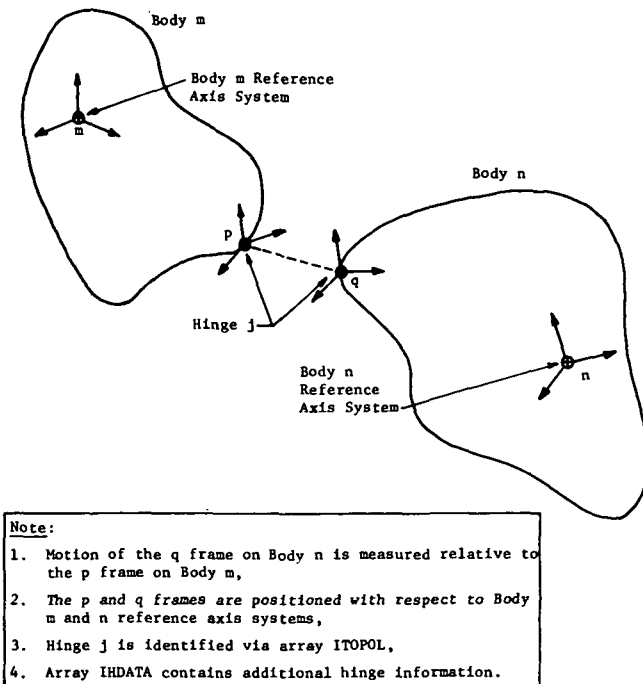
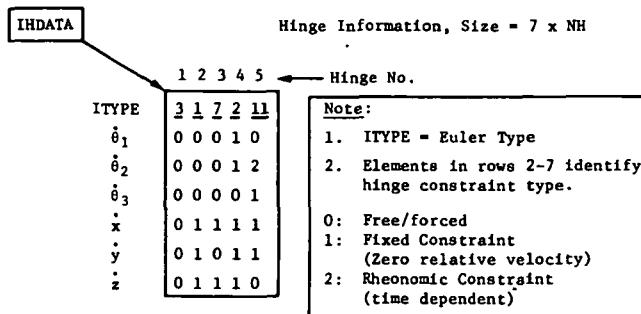
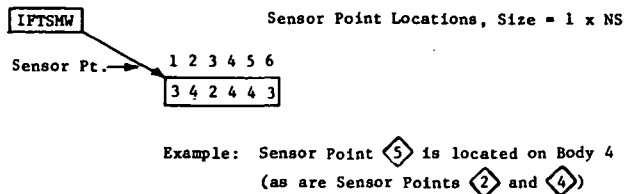
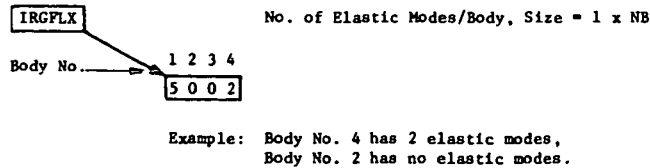
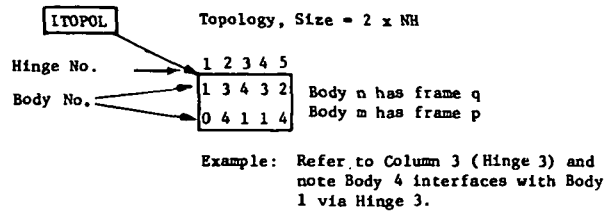


Figure 1. Simulation nomenclature (Sheet 1 of 4).

PROGRAM SIMULATION INTEGER ARRAYS



Additional Remarks:

1. The number of Betas in the state vector equals the sum of "zeros" plus the sum of the "twos" (excluding row 1) in the array.
2. The number of Lambdas (constraints) equals the number of "nonzeros" (excluding row 1) in the array.

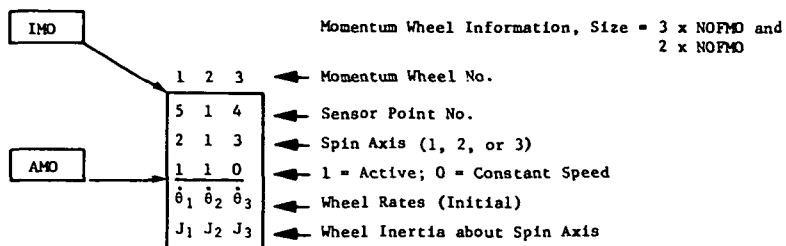
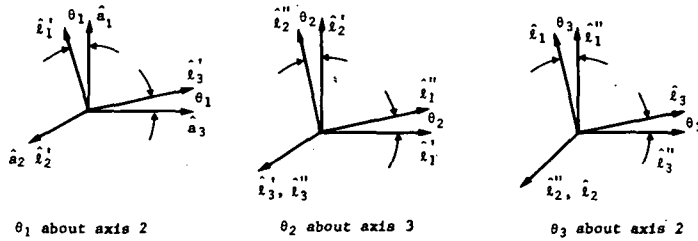


Figure 1. Simulation nomenclature (Sheet 2 of 4).

EULER ANGLE PERMUTATION CANDIDATES ~ ITYPE

	ITYPE =	1	2	3	4	5	6	7	8	9	10	11	12
1st Rotation	Axis of θ_1	1	1	1	1	2	2	2	2	3	3	3	3
2nd Rotation	Axis of θ_2	2	2	3	3	3	3	1	1	1	1	2	2
3rd Rotation	Axis of θ_3	3	1	1	2	1	2	2	3	2	3	3	1

Example: Using ITYPE = 6



$$\begin{Bmatrix} \hat{l}_1 \\ \hat{l}_2 \\ \hat{l}_3 \end{Bmatrix} = \begin{bmatrix} C_3 & 0 & -S_3 \\ 0 & 1 & 0 \\ S_3 & 0 & C_3 \end{bmatrix} \begin{bmatrix} C_2 & S_2 & 0 \\ -S_2 & C_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} C_1 & 0 & -S_1 \\ 0 & 1 & 0 \\ S_1 & 0 & C_1 \end{bmatrix} \begin{Bmatrix} \hat{a}_1 \\ \hat{a}_2 \\ \hat{a}_3 \end{Bmatrix}$$

also

$$\hat{l}_i = {}^1R_j \hat{l}_j$$

CONSOLIDATION OF KINEMATICAL COEFFICIENTS

(The "b" Coefficients)

Body No. Hinge No.	1	2	3	4		
1	q				$\{U\}_1$	$\{\dot{\theta}\}_1$
2			q	p	$\{U\}_2$	$\{\dot{\theta}\}_2$
3	p			q	$\{U\}_3$	$\{\dot{\theta}\}_3$
4	p		q		$\{U\}_4$	$\{\dot{\theta}\}_4$
5		q		p		

Note:

- Only a single entry is in partition for Hinge 1 and it is a q type partition from the connection kinematical array above,
- All other row partitions have both a p and a q constituent for each "hinge" between contiguous bodies.

Figure 1. Simulation nomenclature (Sheet 3 of 4).

CONNECTION KINEMATICS ~ TYPICAL HINGE

$$\begin{array}{c} \text{a "p" Point} \end{array} \quad \begin{array}{c} \text{a "q" Point} \end{array}$$

$$\begin{Bmatrix} \Delta \dot{\theta}_1 \\ \Delta \dot{\theta}_2 \\ \Delta \dot{\theta}_3 \\ \Delta \dot{x}_1 \\ \Delta \dot{x}_2 \\ \Delta \dot{x}_3 \end{Bmatrix} = \begin{Bmatrix} -\pi^{-1} \begin{smallmatrix} R \\ q \end{smallmatrix} \begin{smallmatrix} p \\ p \end{smallmatrix} \begin{smallmatrix} R \\ m \end{smallmatrix} & -\pi^{-1} \begin{smallmatrix} R \\ q \end{smallmatrix} \begin{smallmatrix} p \\ p \end{smallmatrix} \begin{smallmatrix} R \\ m \end{smallmatrix} \sigma_p & +\pi^{-1} \begin{smallmatrix} R \\ q \end{smallmatrix} n & +\pi^{-1} \begin{smallmatrix} R \\ q \end{smallmatrix} n \sigma_q \\ -\begin{smallmatrix} R \\ p \end{smallmatrix} \begin{smallmatrix} S \\ m \end{smallmatrix} p & -\begin{smallmatrix} R \\ p \end{smallmatrix} \begin{smallmatrix} h \\ p \end{smallmatrix} & \begin{smallmatrix} R \\ p \end{smallmatrix} \begin{smallmatrix} q \end{smallmatrix} \begin{smallmatrix} R \\ n \end{smallmatrix} \begin{smallmatrix} S \\ n \end{smallmatrix} q & \begin{smallmatrix} R \\ p \end{smallmatrix} \begin{smallmatrix} q \end{smallmatrix} \begin{smallmatrix} R \\ n \end{smallmatrix} \begin{smallmatrix} h \\ q \end{smallmatrix} \end{Bmatrix} \begin{Bmatrix} \dot{\theta}_n \\ \dot{\theta}_m \end{Bmatrix}$$

Note:

- ${}_i R_j$ is a rotation transformation from the j to the i reference frame.
- In general, ${}_i R_j {}_j R_k = {}_i R_k$ and π^{-1} is a transformation relating Euler rates to body axes projections of the angular velocity vector.
- σ_i is a modal slope matrix in the i frame,
- h_i is a modal deflection matrix in the i frame,
- S_{ij} is a skew symmetric matrix whose elements are the components of the vector i to j in the i th frame.

$$\begin{Bmatrix} \dot{\theta}_j \end{Bmatrix} = \begin{Bmatrix} \omega_x \\ \omega_y \\ \omega_z \\ u \\ v \\ w \\ \dot{\xi}_1 \\ \dot{\xi}_2 \\ \vdots \\ \dot{\xi}_n \end{Bmatrix}_j$$

STATE VECTOR ARRANGEMENT

CONSTRAINT FORCE/TORQUE VECTOR

$$\begin{Bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \\ \lambda_5 \\ \lambda_6 \\ \lambda_7 \\ \lambda_8 \\ \lambda_9 \\ \lambda_{10} \\ \lambda_{11} \\ \lambda_{12} \\ \lambda_{13} \\ \lambda_{14} \end{Bmatrix} = \begin{Bmatrix} (F_x)_2 \\ (F_y)_2 \\ (F_z)_2 \\ (F_x)_3 \\ (F_z)_3 \\ (T_1)_4 \\ (T_2)_4 \\ (F_x)_4 \\ (F_y)_4 \\ (F_z)_4 \\ (T_2)_5 \\ (T_3)_5 \\ (F_x)_5 \\ (F_y)_5 \end{Bmatrix}$$

$(F_x)_1$ refers to Hinge 1

$$\begin{Bmatrix} \{U\}_1 \\ \{U\}_2 \\ \{U\}_3 \\ \{U\}_4 \\ \{\xi\}_1 \\ \{\xi\}_2 \\ \{\xi\}_3 \\ \{\xi\}_4 \\ \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \\ \dot{\theta}_5 \\ \dot{\theta}_6 \\ \dot{\theta}_7 \\ \dot{\theta}_8 \\ \dot{\theta}_9 \\ \dot{\theta}_{10} \\ \dot{\theta}_{11} \\ \dot{\theta}_{12} \\ \dot{\theta}_{13} \\ \dot{\theta}_{14} \\ \dot{\theta}_{15} \\ \dot{\theta}_{16} \\ \dot{\theta}_{17} \\ \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{Bmatrix} ; \begin{Bmatrix} \{\dot{U}\}_1 \\ \{\dot{U}\}_2 \\ \{\dot{U}\}_3 \\ \{\dot{U}\}_4 \\ \{\dot{\xi}\}_1 \\ \{\dot{\xi}\}_2 \\ \{\dot{\xi}\}_3 \\ \{\dot{\xi}\}_4 \\ \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \\ \dot{\theta}_5 \\ \dot{\theta}_6 \\ \dot{\theta}_7 \\ \dot{\theta}_8 \\ \dot{\theta}_9 \\ \dot{\theta}_{10} \\ \dot{\theta}_{11} \\ \dot{\theta}_{12} \\ \dot{\theta}_{13} \\ \dot{\theta}_{14} \\ \dot{\theta}_{15} \\ \dot{\theta}_{16} \\ \dot{\theta}_{17} \\ \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{Bmatrix}$$

where $\{U\}_j = \begin{Bmatrix} \omega_x \\ \omega_y \\ \omega_z \\ u \\ v \\ w \\ \dot{\xi}_1 \\ \dot{\xi}_2 \\ \vdots \\ \dot{\xi}_n \end{Bmatrix}_j$

For this example, prescribed as function of time, i.e., a rheonomic constraint as noted in IHDATA

Figure 1. Simulation nomenclature (Sheet 4 of 4).

know where certain variables are located so that he can couple the control law into the simulation, and, secondly, the user must know the order of the state variables in order to interpret results. Figure 1 presents the state variable order consistent with the illustrative problem and other related information. The state variables shown represent a typical arrangement in that all of the various types of variables resulting from the multiple options available within the simulation are present. The order of the constraints, λ , is also noted. Note that, although the user introduces the control variables into the state vector, these variables, δ , will always appear after the betas, β . Furthermore, the user may also introduce auxiliary variables (plant sensor signals and control-system outputs) for use in the linearized studies. These auxiliary variables should be placed (by the user) after the control variables, δ , and in the order: plant sensor signals, X_{ss} , followed by control-system outputs, B .

C. General Use Information

A number of guidelines must be followed in setting up a particular simulation. Some were detailed previously and are concisely summarized here:

- There must be at least two bodies; a single-body problem is simulated by including a dummy body that is not connected to the body to be analyzed.
- Body 1 is always positioned relative to the inertial reference.
- Bodies are numbered from 1 to NB in an arbitrary order.
- Each body (except body 1) must have at least *one* hinge; body 1 must have at least two hinges.
- Hinges are numbered from 1 to NH in an arbitrary order, but hinge 1 is, by definition, the hinge on body 1 between body 1 and the inertial origin; hence, hinge 1 can only appear on body 1.
- There must be at least one sensor point for a given simulation.
- Sensor points are numbered from 1 to NS in an arbitrary manner.
- A typical flexible body requires mass and modal data that reflects a coordinate system that is consistent with the body axis reference system for that body (e.g., a modal coupling approach that establishes modal properties for a given body would have to use the same reference body axis system).
- For frequency-domain studies, only as many control output variables can be identified for introducing into the state equations as there are control-system variables to begin with. Similarly, no more sensor-signal variables can be identified than plant variables that appear in the original independent state equations.
- The user must ensure that the user-supplied package has dimensions consistent with NHMAX for the following arrays:
SK(NSK, NHMAX), DK(NDK, NHMAX), and HNGT(NHT, NHMAX)

where

NHMAX = dimensioned maximum number of hinges

NSK = 3 or 6, depending on the nature of hinge freedom

NDK = 3 or 6, depending on the nature of hinge freedom

NHT = 3 or 6, depending on the nature of hinge freedom

and, if rotation only, then $NSK = NDK = NHT = 3$; if rotation and translation, then $NSK = NDK = NHT = 6$.

- The inertial properties of all the momentum wheels in a particular body must be included in that body's inertia description (whether rigid or flexible) because inertial coupling is used.

II. PROGRAM SEGMENTATION

The digital code has been segmented into an executive overlay that governs the succeeding program flow and four supporting primary overlays, each with a separate and dedicated purpose. The basic program flow is depicted in figure 2.

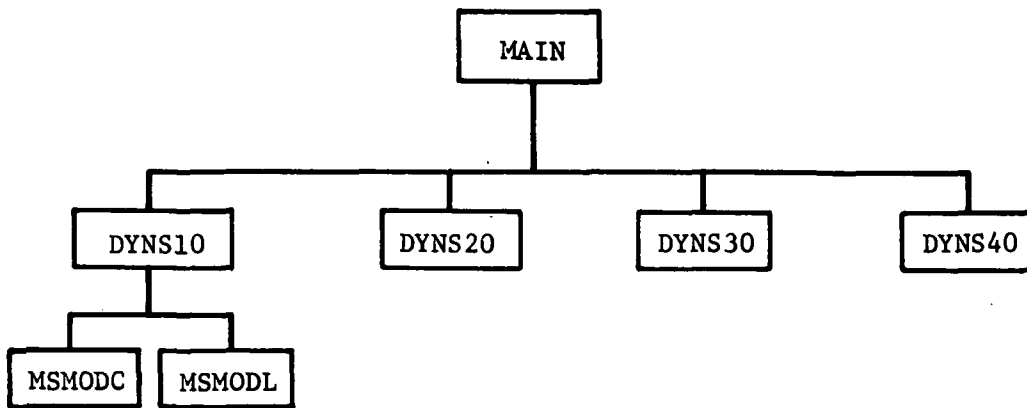


Figure 2. DISCOS program segmentation.

Table 1 summarizes the intended purpose of the fundamental components in the program structure.

The executive overlay (MAIN) initiates the simulation by reading job-identification information and then passes control to the first primary overlay (DYNS10), which represents the basic data input segment. This overlay may be viewed as the program segment that builds the model from the input data. A series of topology checks are made as the data are loaded within this overlay to better ensure proper modeling of the physical system. This overlay utilizes two additional secondary overlays for processing certain types of inertial and modal data.

Table 1
DISCOS Overlay Description

Segment Name	Purpose
Primary Overlays	
MAIN	Executive program control
DYNS10	Data input
DYNS20	Simulation of problem, linearization of state equations, and nonlinear time response
DYNS30	Plot results from nonlinear or linearized time response
DYNS40	Frequency-domain analysis, linearized time response, and frequency-domain displays (Bode, Nichols, Nyquist, Root Locus)
Secondary Overlays (called from DYNS10)	
MSMODL	Flexible-body data inputs for lumped mass representation
MSMODC	Flexible-body data inputs for consistent mass representation

After overlay DYNS10 has structured the basic data for simulation, control is returned to the executive overlay that, in turn, passes control to the second overlay, DYNS20. This overlay performs the actual problem mechanization and develops the nonlinear formulation that is the foundation for the entire dynamic simulation program.

During a given simulation, the executive overlay always calls both the first and the second primary overlays, DYNS10 and DYNS20, but, depending on certain input control parameters, may or may not call the time-history plot overlay, DYNS30, or the linearized system-analysis overlay, DYNS40.

Simulation of a particular problem has its basis within the algorithms contained in the program subroutine, YDOT, which establishes the canonical first-order differential equations that govern the dynamical motion. In turn, this routine addresses another subprogram, TORQUE, which, in turn, activates the user-supplied modules that relate to the particular simulation being considered. The following section describes these modules in further detail.

III. DELINEATION OF USER-SUPPLIED MODULES

The program has been written under the assumption that certain user-supplied modules are available for completing a given problem. In this manner, the user has considerable latitude concerning how certain particulars related to a given simulation should be handled. Control-law specification, external torque inputs, and identification of plant sensor signals and

control-system outputs are examples of items handled by the user. With this concept in mind, several subprograms have been placed under user control, but with certain restrictions and guidelines to which the user must adhere. Later comments will identify certain requirements associated with these user-supplied modules.

A. Logic Flow

It is worthwhile to consider a flow-chart segment of the program (figure 3) and its chronology within the solution process. The order in which the user-supplied modules are called is indicated by integers 1 through 7 for subroutines and 8 and 9 for functions.

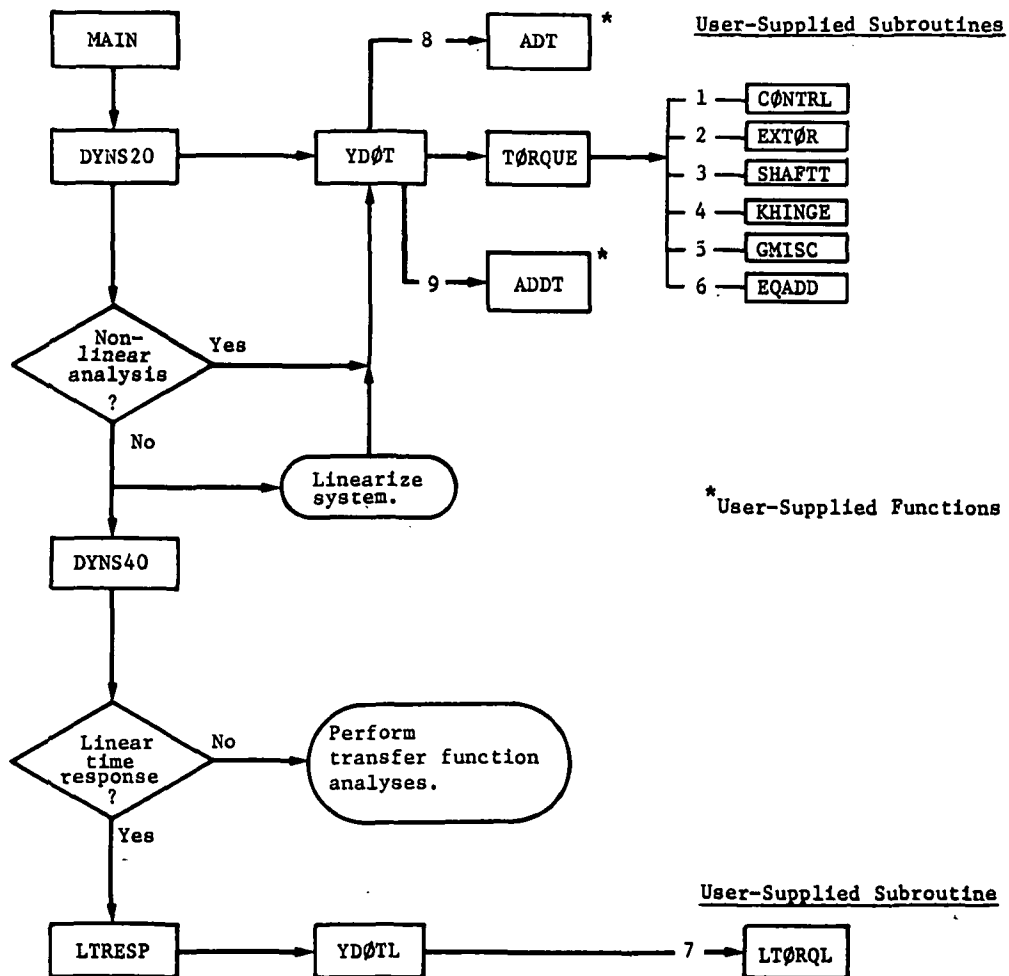


Figure 3. Chronology of addressing user-pack.

B. User-Supplied Module Specifics

Each user-supplied subprogram has specific intended purposes and has been coded to fulfill these goals. The user can extend the scope of any of these modules with his own code, but these routines must perform certain tasks. Because the potential user should be familiar with many details of the user-supplied package, the following paragraphs are devoted to each of the user-supplied modules. Because reference will be made to some of the programming logic contained in the DISCOS subroutine TORQUE, this logic has been put into flow-chart form as figure 4. Additional comments and rules to be followed in coding the user modules can be found in the comment cards included in the program listing of the default form of each of the routines and in subroutine DEF5.

1. CONTRL

CONTRL is the first of the user-supplied subroutines and is always called by DYN20. The primary purpose of CONTRL is to establish the time derivatives of the control-system variables. These variables may be required by some of the other user routines that are activated after CONTRL has been addressed. The routine must also establish the number of plant sensor signals, NXSS, and the number of control-system outputs, NBTQ, that are transmitted through the common block, /LDSIZE/,* to the remainder of the program. For transfer-function studies, the user must also identify whether or not transfer-function polynomials are to be used. This is accomplished in a data statement (in CONTRL) with the variable, NPLY, which is the number of polynomial ratio pairs (numerator and denominator) to be used. For $NPLY \neq 0$, the first call to CONTRL will read in the polynomial coefficients.

Subroutine CONTRL contains a good deal of information pertaining to the simulation by virtue of its common blocks. Section III-C identifies the constituents of the common blocks contained in this and other modules. The program user can establish additional common blocks to transfer information between the separate user-pak modules.[†]

2. EXTOR

Subroutine EXTOR establishes the external system torques. Typically, this module can accommodate items such as reaction-control system (RCS) forces and torques, aerodynamics, and/or solar wind. The user can also extend this routine to include other state-dependent torques. In summary, EXTOR can be used as a "catchall" for including any additional forces and torques that affect the system. A single call to EXTOR from subroutine TORQUE establishes an integer array, ISNP, whose elements identify which sensor points are to be used for force/torque inputs. A vector containing torque and force components (ordered: $T_x, T_y, T_z, F_x, F_y, F_z$) is then established for each of the force/torque sensor points and is

*Internal logic loops require that NXSS be set greater than zero.

†See subroutine DEF1 and DEF5.

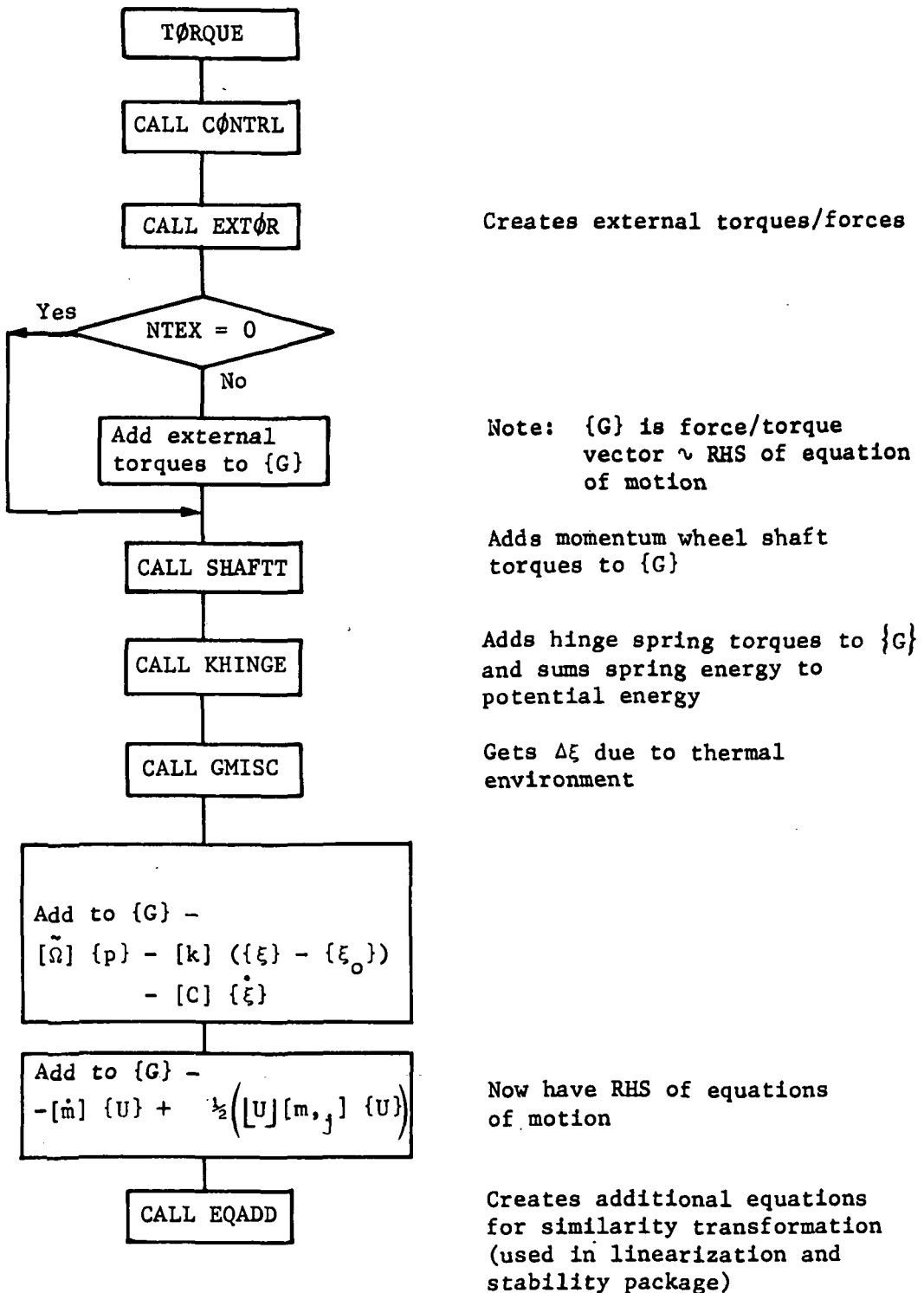


Figure 4. Subroutine TORQUE flow diagram.

placed as a column into the array, TEX. The vector of discrete forces and torques (referred to the local sensor-axis system) is returned to subroutine TORQUE from EXTOR. These forces and torques are then transformed and added to the total system external force/torque array $\{G\}$. The user can bypass EXTOR-related calculations by setting the variable, NTEX, equal to zero.

3. SHAFTT

Subroutine SHAFTT establishes the shaft torque for each of the nonconstant-speed momentum wheels. Zeros are inserted for the torque contributions to the external torque vector for a constant-speed wheel.

4. KHINGE

Subroutine KHINGE sets up hinge-spring and dashpot torques/forces and accounts for potential energy contributions caused by hinge-spring deflections. The user must identify where spring rates and dashpot constants should be found. This can be easily handled by a user-specified equivalence statement within subroutine KHINGE to locate the leading stiffness and damping elements within the data block identified as CNTDTA. Note that subroutine KHINGE (see subroutine listing) contains statements of the following form:

```
DIMENSION SK(3, NHMAX), DK(3, NHMAX), HNGT(3, NHMAX)
```

```
      .
```

```
      .
```

```
      .
```

```
DO 10 I=1,3
```

```
      .
```

```
      .
```

```
DO 15 I=1,3
```

```
      .
```

```
      .
```

```
DO 20 I=1,3
```

```
      .
```

```
      .
```

where integer 3 reflects the fact that consideration has been restricted to admitting only rotational springs at each hinge. If the user wants to also include springs or dashpots in relative translation at the hinge points, the 3 in the foregoing statements must be changed

to a 6, and appropriate spring rates and/or dashpots included within the data input array, CNTDTA. Further, the equivalence statement that locates the first spring rate, SK(1), and dashpot constant, DK(1), reflect an order that is consistent with the hinge order (i.e., the first three elements in CNTDTA beginning with the location corresponding to the leading element of SK represents in order K_{θ_1} , K_{θ_2} , and K_{θ_3} for the first hinge). A similar relationship exists for the array, DK. For example, in KHINGE, note that:

$$\text{EQUIVALENCE}(\text{CNTDTA}(\text{K}), \text{SK}(1)), (\text{CNTDTA}(\text{L}), \text{DK}(1))$$

and the array, CNTDTA, would be:

$$\begin{array}{lcl} \text{CNTDTA} & = & \dots \underbrace{K_{\theta_1} \ K_{\theta_2} \ K_{\theta_3}}_{\text{Hinge 1 (springs)}} \dots \underbrace{K_{\theta_1} \ K_{\theta_2} \ K_{\theta_3}}_{\text{Hinge NH (springs)}} \dots \\ \text{element (K)} & \nearrow & \\ & & \\ \text{element (L)} & \nearrow & \dots \underbrace{C_{\dot{\theta}_1} \ C_{\dot{\theta}_2} \ C_{\dot{\theta}_3}}_{\text{Hinge 1 (dashpots)}} \dots \underbrace{C_{\dot{\theta}_1} \ C_{\dot{\theta}_2} \ C_{\dot{\theta}_3}}_{\text{Hinge NH (dashpots)}} \dots \end{array}$$

where the order of $\theta_{1,2,3}$ is consistent with the Euler rotation type for the hinge "q" triad, specified by user in array IHDATA.

The remainder of KHINGE is concerned with the proper placement of the spring/dashpot forces and torques onto the composite NB bodies (generalization of forces and torques) and should remain unchanged.

The user can modify the referenced torques and forces immediately after the (DO 10 L=1, NH) loop if he desires, but must be careful to ensure that the proper force or torque is correctly applied to accomplish the desired result.

5. GMISC

Subroutine GMISC is reserved for implementing torque/force contributions from thermal gradient effects. The entire state vector, along with component position and attitude information, is available by means of transfer through labeled common arrays. Paragraph III.C provides more detail on the information contained in these common blocks.

6. EQADD

Subroutine EQADD establishes additional equations for use in the linearized time-domain analyses. It must identify the number of additional equations introduced by the variable, NAUX (number of auxiliary equations): These equations relate plant sensor signals, X_{ss}^i , and control-system output forces/torques, B^i , to the system state in the specified order.

The additional variables will be automatically placed in the state vector and become an integral part of system transfer-function evaluations. (See comment cards for subroutines EQADD and DEF5.)

7. LTORQL

Subroutine LTORQL establishes the $b_{ik} U^k$ portion of the right-hand side of

$$\dot{z}^i = A_{ij} z^j + b_{ik} U^k$$

which is used for the linearized time response. This corresponds to the external excitations for the transformed variables, z^i , leading to evaluation of the perturbation response.

8. ADT (Function)

Function ADT is used in conjunction with subfunction ADDT to implement prescribed kinematical motion in the hinge coordinates. With reference to figure 1, the array, IHDATA(I,J), $I \geq 1$, may have 2 as an entry, indicating that the J^{th} hinge has velocity and acceleration prescribed in that coordinate. The argument of this subfunction is: ADT(IC,T) with $IC=6*(J-1)+(I-1)$ corresponding to $IHDATA(I,J)=2$. The integer, IC, and the time, T, are passed into ADT via argument by the calling subroutine so that the velocity, $\dot{\alpha}$, may be established for the proper hinge coordinate as a function of time.

For a given rheonomic constraint, note that both $\dot{\alpha}$ and $\ddot{\alpha}$ must be set by subfunction. It is conceivable that the user knows $\ddot{\alpha}$ as the exact mathematical time derivative of $\dot{\alpha}$. It would seem that the natural thing to do is to create ADT and ADDT subfunctions to return consistent $\dot{\alpha}$ and $\ddot{\alpha}$, respectively. This is *not* the best thing to do, however, because of numerical integration characteristics. The numerical integration of $\{\dot{U}\}$ reflects the use of $\ddot{\alpha}$. The resulting $\{U\}$ reflects a numerically integrated $\dot{\alpha}$ that cannot be consistent with a value obtained by any method other than numerical integration. The consequences of this are seen as slight errors in motion response and a large spurious change in system momenta.

The best way to effect rheonomic constraints is to use values of $\dot{\alpha}$ obtained from numerically integrating $\ddot{\alpha}$. This can be easily done by using additional differential equations that are accommodated in the state vector as additional "control variables" or $\{\delta\}$. Thus, when all actual control variable rates are established (in subroutine CONTRL), the user need only code additional expressions to set $\dot{\delta}$ (additional) = $\ddot{\alpha}$ (desired). The statements within subfunction ADT merely return $ADT=Y(K)$; the state vector, Y, is available in labeled common /VECTOR/, and K corresponds to the location in Y where the $\delta = \dot{\alpha}$ control variable resides. Of course, IC must be tested so that the appropriate $\dot{\alpha} = \delta$ is returned.

9. ADDT (Function)

Function ADDT is discussed with regard to its relationship to ADT (paragraph III.B.8). This function has arguments, ADDT(IC,T), identical to those of ADT, and returns values of α for appropriate IC and T consistent with the $\dot{\alpha}$ returned by ADT. Note that, in figure 3, the chronology is such that subroutine CONTRL is addressed before function ADDT so that CONTRL can establish a value of δ (additional) = $\ddot{\alpha}$ (desired) to put in the state-vector time derivative, YDT (also available in labeled common/VECTOR/). Now, for the appropriate time, T, and IC, it is necessary only to set ADDT=YDT(K), where K again corresponds to the location in Y at which the $\delta = \ddot{\alpha}$ auxiliary control variable resides.

C. Selected Common-Block Information

To compute specific variables required for the user-supplied modules, the program user will often require certain information that is calculated and stored within the program. Such information about the simulation is stored in multidimensional array form within labeled common blocks. These data provide a good supplement to the state variable content that has been discussed previously in that the user can extract both total and relative positions and rates for any component of the simulated dynamical system if he has a firm understanding of where certain data reside within the program. The following subsections discuss selected common-block arrays to familiarize the potential user with their content.*

1. IBHBSRDI

Common block /BHBSRD/ contains three separate groups of information that the user may require. This information is concisely summarized in double- and/or triple-subscripted arrays as

BS(6,6+NMDBOD,NSPMAX)

ROL(3,3,NBMAX)

DOL(3,NBMAX)

where the following items are noted:

NMDBOD = maximum-dimensioned number of modes per body

NBMAX = maximum-dimensioned number of bodies

NSPMAX = maximum-dimensioned number of sensor points

The array, BS(i,j,k), contains the kinematical coefficients for all of the "sensor" points. The rows (subscript i=1,2...6) of the array refer to (in order: ω_x , ω_y , ω_z , u, v, w) the components of absolute angular and translational velocity (sensor referenced) at sensor

*Also see subroutines DEF1 and DEF5.

point k. The columns of the array (subscript j) refer to the $j = 1, 2, \dots, 6 + \text{number of elastic modes on body containing sensor point k}$. Thus, in general, the following expression can be used to determine the projection (the i^{th} velocity component) onto the triad located at sensor point k:

$$Vel_i = BS(i, j_1, \dots, j_L, k) \cdot \tilde{U}^j$$

The array, $ROL(i,j,k)$, contains the rotation transformations relating the body axis systems to the inertial reference. The elements of the array are the direction cosines between the body axes, e_k , and the fixed inertial system, e_o . Subscript k denotes the body number.

The array, $DOL(i,k)$, contains the three vector components, X, Y, Z, from the inertial reference to the body axis system, e_k , for each body.

2. /SPECIF/

Common block /SPECIF/ contains information that the user may require. These arrays are:

BETAH(6,NHMAX)

BETAHD(6,NHMAX)

RS(3,3,2*(NSPMAX))

DS(3,2*(NSPMAX))

where the following items are noted:

NHMAX = maximum-dimensioned number of hinges

NSPMAX = maximum-dimensioned number of sensor points

The arrays, $BETAH(i,j)$ and $BETAHD(i,j)$, contain the hinge BETA's and rates for hinge j, respectively. The order (i subscript) is given as

$$\begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\Delta}_1 \\ \dot{\Delta}_2 \\ \dot{\Delta}_3 \end{bmatrix}$$

where $\dot{\theta}_i$ is the i^{th} Euler angle rate consistent with ITYPE for hinge j, and $\dot{\Delta}_i$ is the i^{th} velocity component of point q relative to point p in the p frame for hinge j.

The array, RS(i,j,k), contains the rotation transformations (direction cosines) between the sensor-point axis system and the body axis system (body on which sensor is located). Two sets of transformations are identified for a given sensor point. The first represents misalignment of the two triads without elastic deformation, and the second includes the elastic deformation. The ordering (subscript k) proceeds as follows: The ℓ^{th} sensor rotation (without elastic deformation) is located at $k = 2*\ell - 1$. The total rotation transformation for the ℓ^{th} sensor is located at $k = 2*\ell$. For a rigid body, these two transformations are identical.

The array, DS(i,k), contains the three components of the vector from the body axis system to the body sensor points (in the body axis system). Here again, there are two sets of vectors (rigid body and rigid body + elastic) for each sensor point. The first is for rigid body and the second includes the elastic deformation. The addressing algorithm is the same as that for RS(i,j,k).

IV. PROGRAM INPUTS

The dynamic simulation program utilizes some basic data input subroutines in an attempt to standardize a large amount of the bulk data input. Additional formatted inputs have been used when it is more meaningful (and more efficient) to do so. As stated in the following section, there is a large amount of data input via subroutines READ and READIM. Therefore, it is useful to familiarize the reader with these two routines before describing overall program data input requirements.

A. Subroutines READ and READIM

Subprograms READ and READIM are structured to read matrix arrays in floating-point (real) notation (subroutine READ) or fixed-point (integer) notation (subroutine READIM). A thorough discussion of the routines and their supporting subroutines is contained in Appendix A. The following discussion gives a cursory overview of their use.

The routines are activated by a FORTRAN call of either of the following forms:

CALL READ (A, NR, NC, KR, KC)

CALL READIM (IA, NR, NC, KR, KC)

where the arguments in the call statement are:

A, (IA) = floating (fixed) matrix array of size NR by NC

NR = number of rows in array

NC = number of columns in array

KR = row dimension of array in calling program

KC = column dimension of array in calling program

A call to either of these input routines requires that the data be in the following formats:

- Subroutine READ

First card—Matrix name, NR, NC with format (A6, I4, I5)

Middle cards—Data with format (2I5, 4D17.8)

First I5 is row number

Second I5 is column number of leading D17.8 field

Next 4D17.8 are elements of the array

Last card—Ten zeros in columns 1 through 10

- Subroutine READIM

First card—Matrix name, NR, NC with format (A6, I4, I5)

Middle cards—Data with format (2I5, 14I5)

First I5 is row number

Second I5 is column number of leading I5 field

Next 14I5 are elements of the array

Last card—Ten zeros in columns 1 through 10

B. Input Data Stream

This section presents the program system input data stream, together with the data input control logic. The approach herein is to first introduce an overview of the data inputs and program control logic in the form of a flow diagram (figure 5) and to then identify the details in much the same way as the FORTRAN code accepts the data inputs. This method of presentation has been chosen because it most closely relates to the actual processing of the user inputs for a given simulation. In addition, the user can follow the program control or switching logic to determine just what data are required for completing a particular simulation.

To keep this documentation current with the program, subroutine DEF3 will contain a detailed description of the input data. As new capability and data input requirements are added to DISCOS, documentation will be kept current by the insertion of suitable comment cards in subroutine DEF3.

V. PROGRAM OUTPUTS

This section discusses the various program output information and correlates the output data with both the input data and the problem simulation. This information is presented in

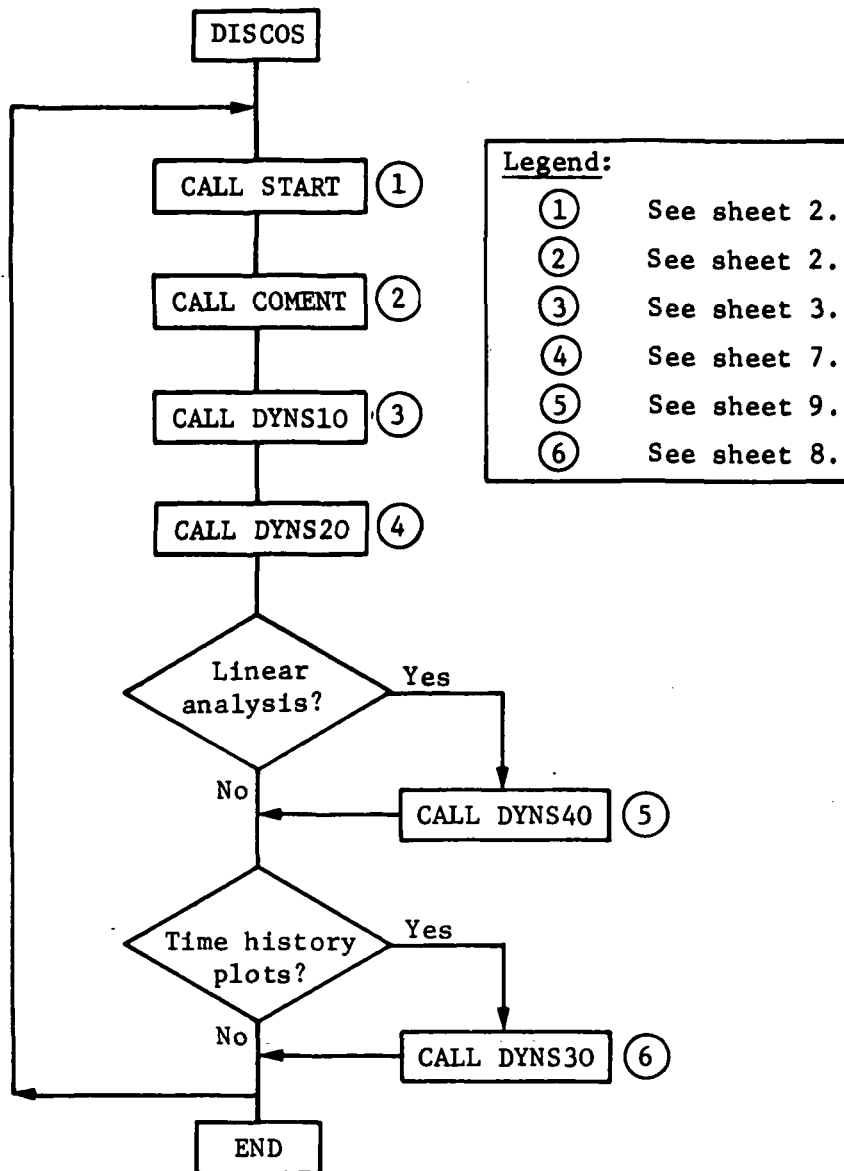


Figure 5. Program system DISCOS data stream flow (Sheet 1 of 12).

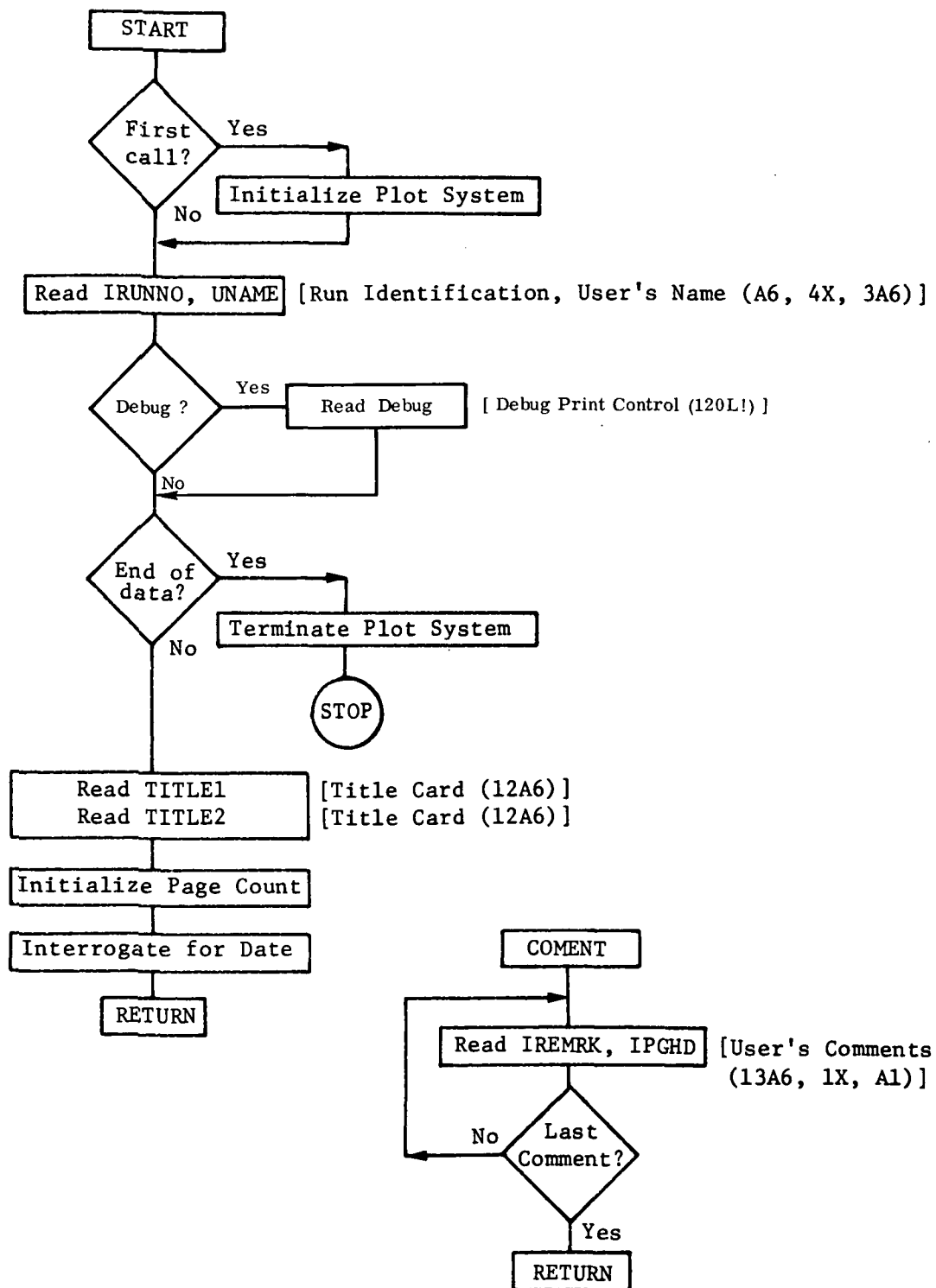


Figure 5. Program system DISCOS data stream flow (Sheet 2 of 12).

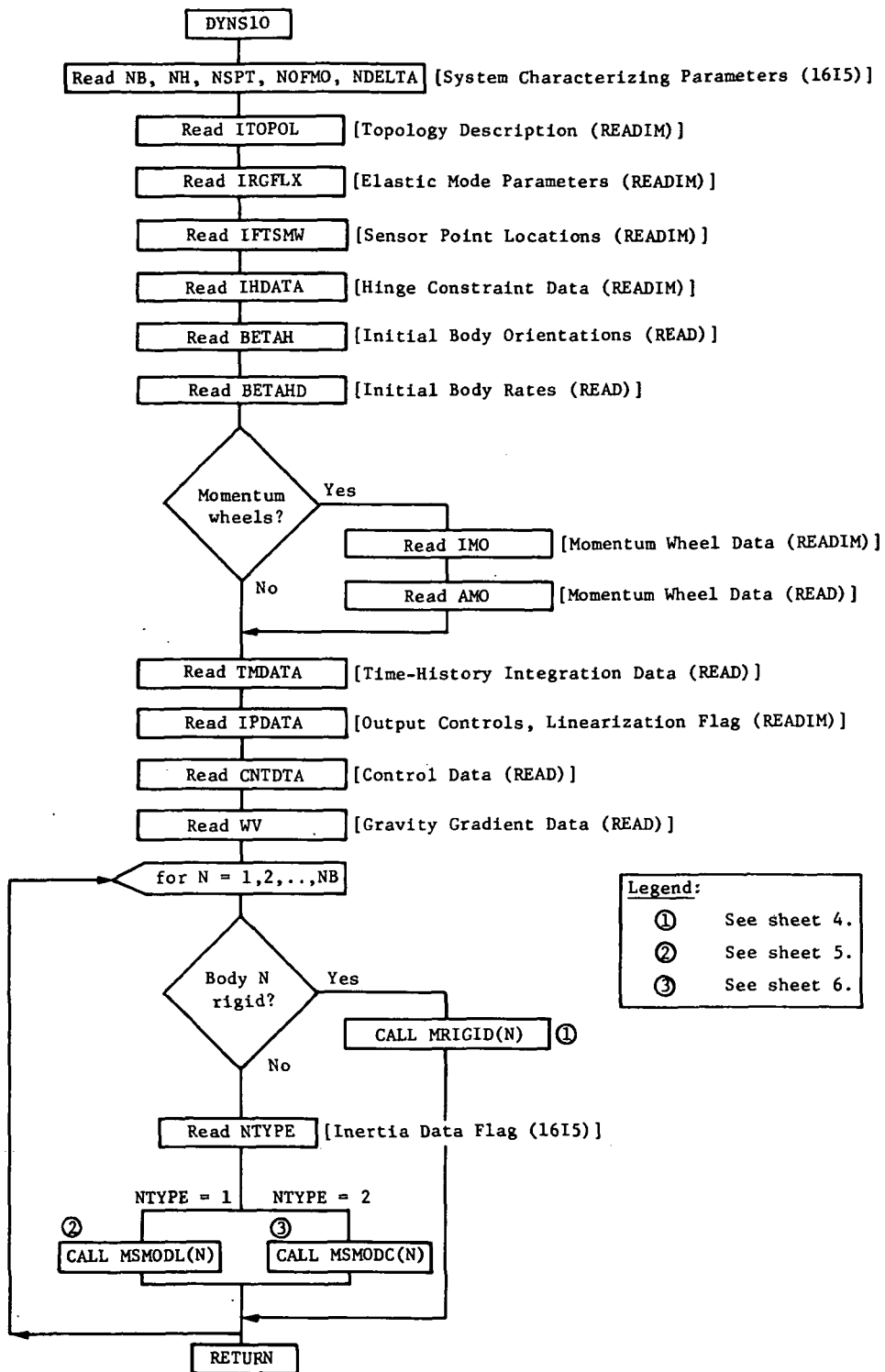


Figure 5. Program system DISCOS data stream flow (Sheet 3 of 12).

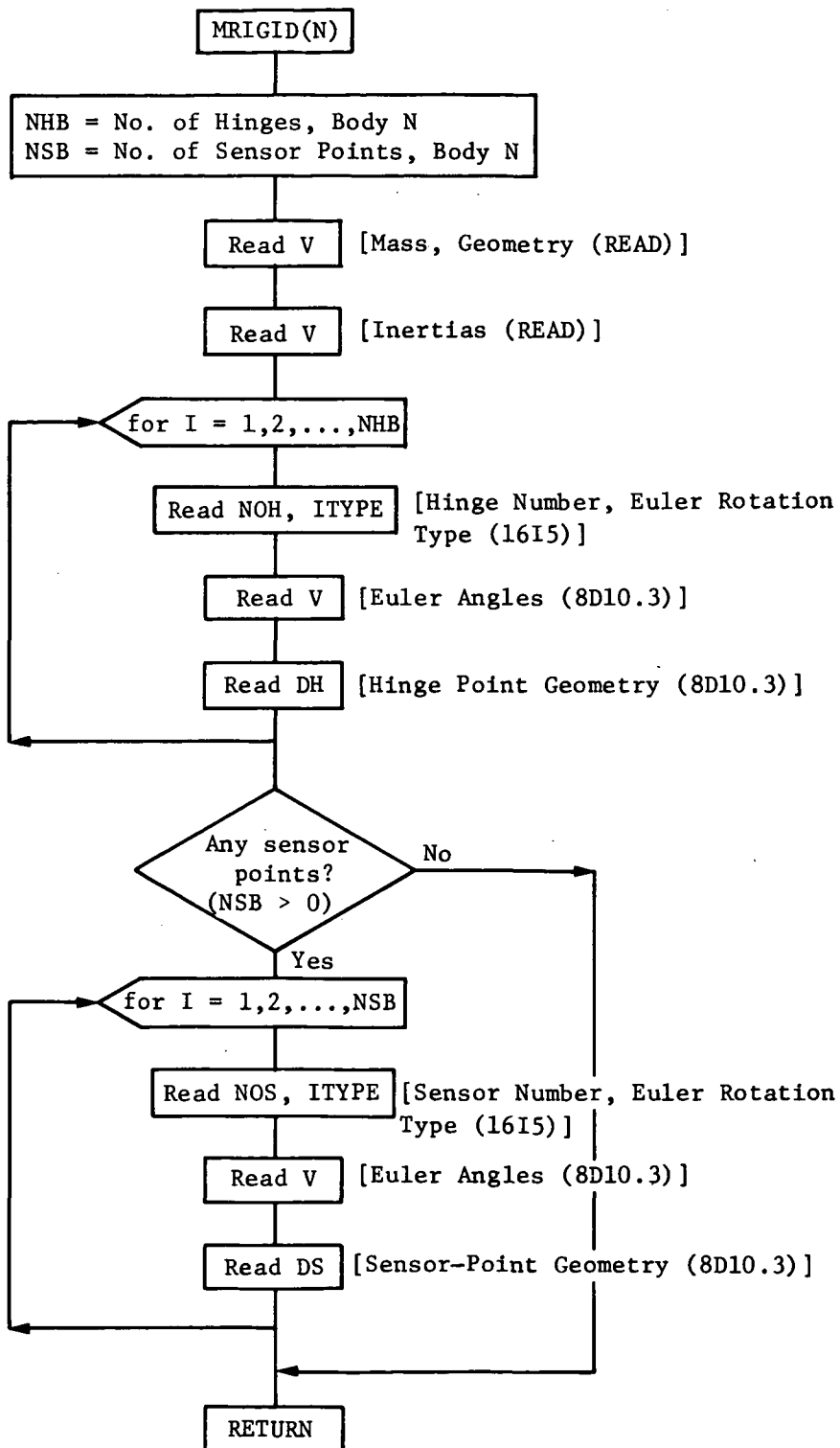


Figure 5. Program system DISCOS data stream flow (Sheet 4 of 12).

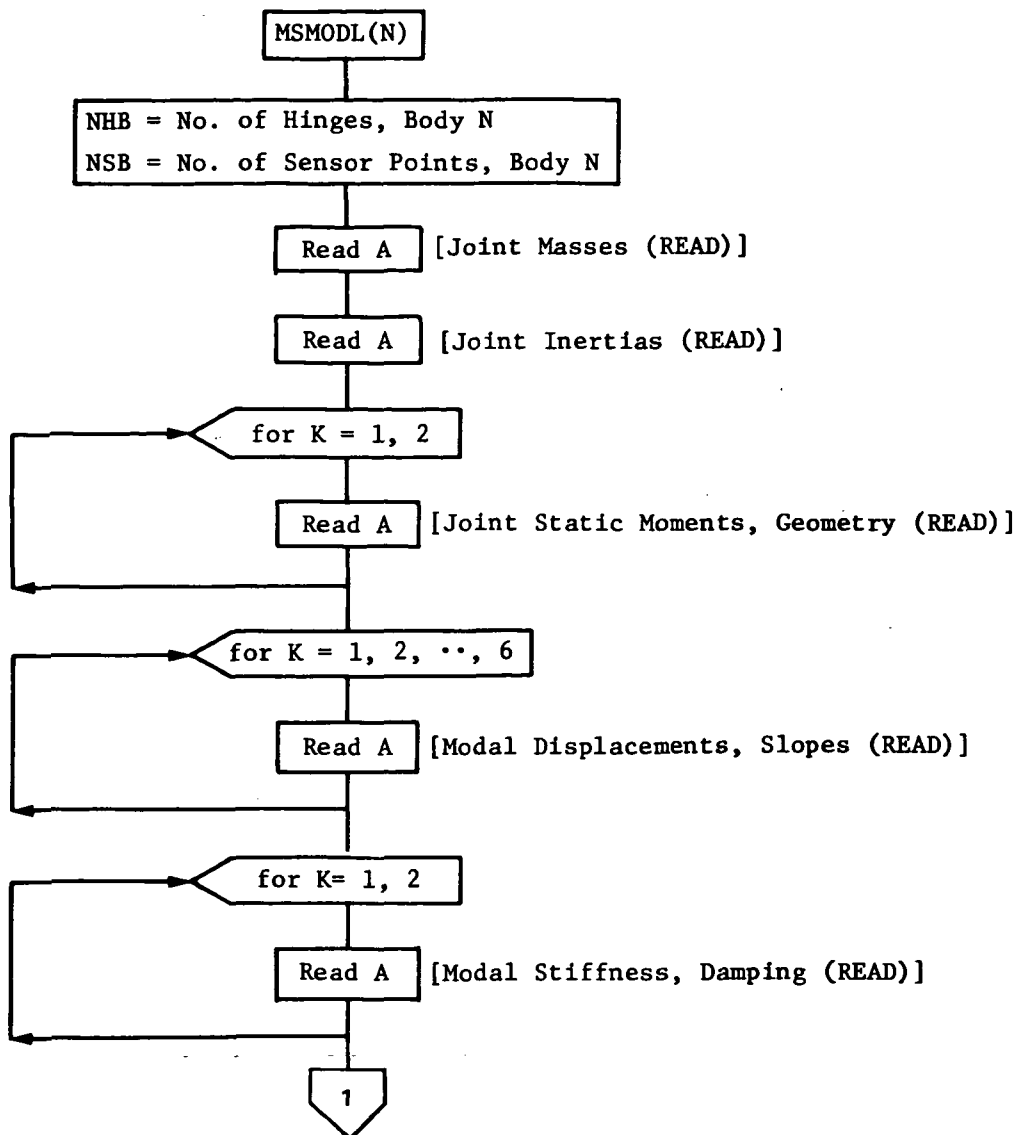


Figure 5. Program system DISCOS data stream flow (Sheet 5 of 12).

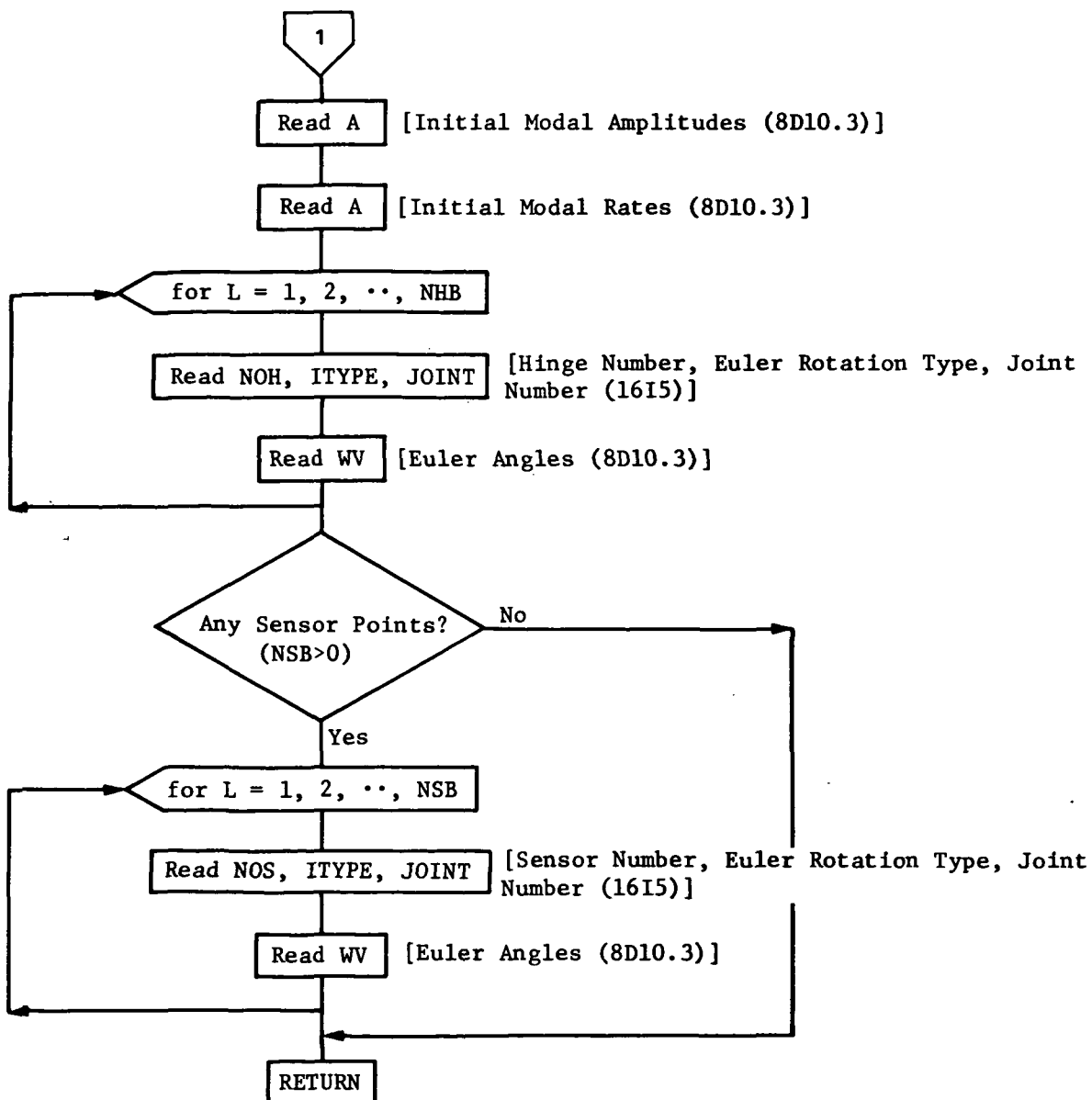


Figure 5. Program system DISCOS data stream flow (Sheet 6 of 12).

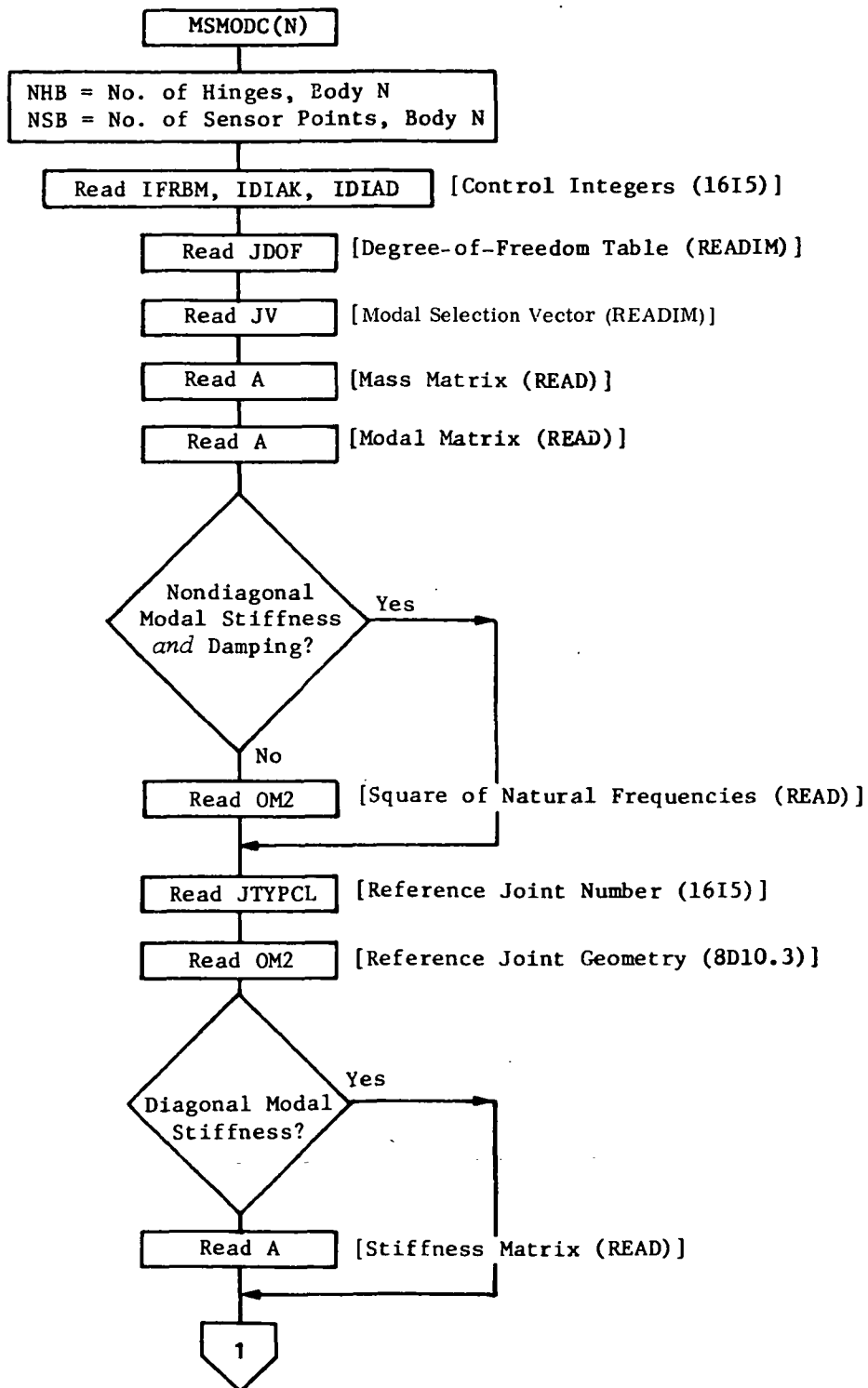


Figure 5. Program system DISCOS data stream flow (Sheet 7 of 12).

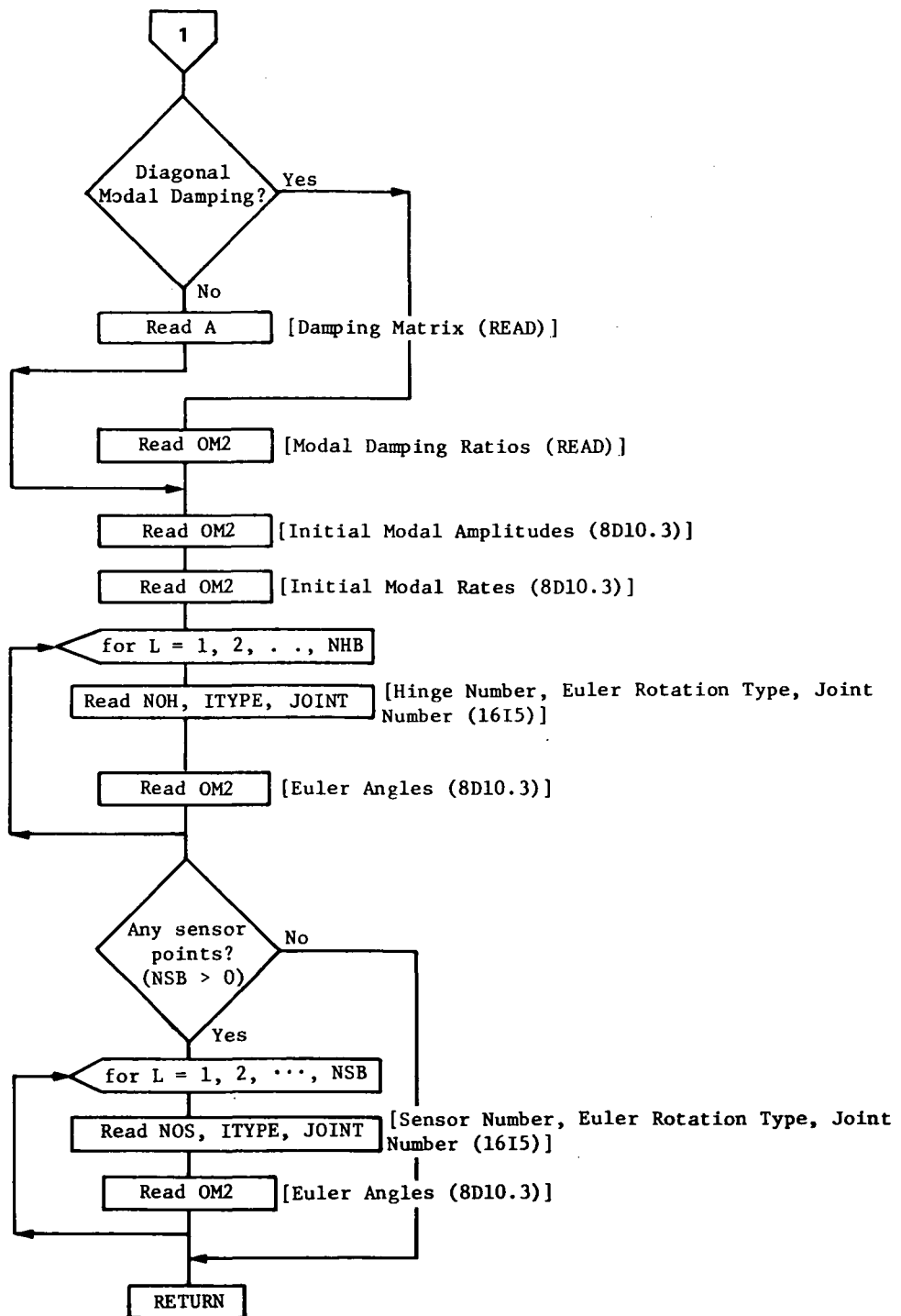


Figure 5. Program system DISCOS data stream flow (Sheet 8 of 12).

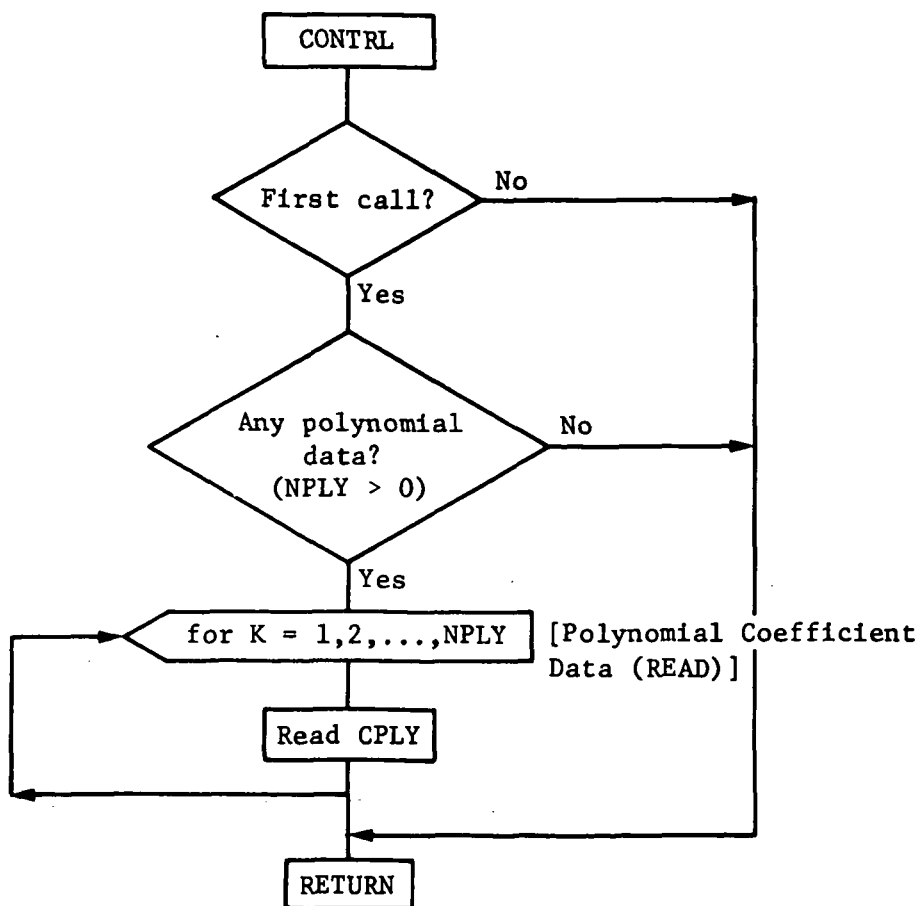
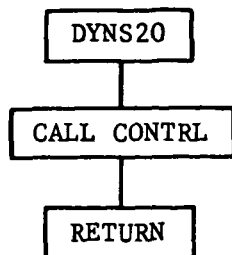


Figure 5. Program system DISCOS data stream flow (Sheet 9 of 12).

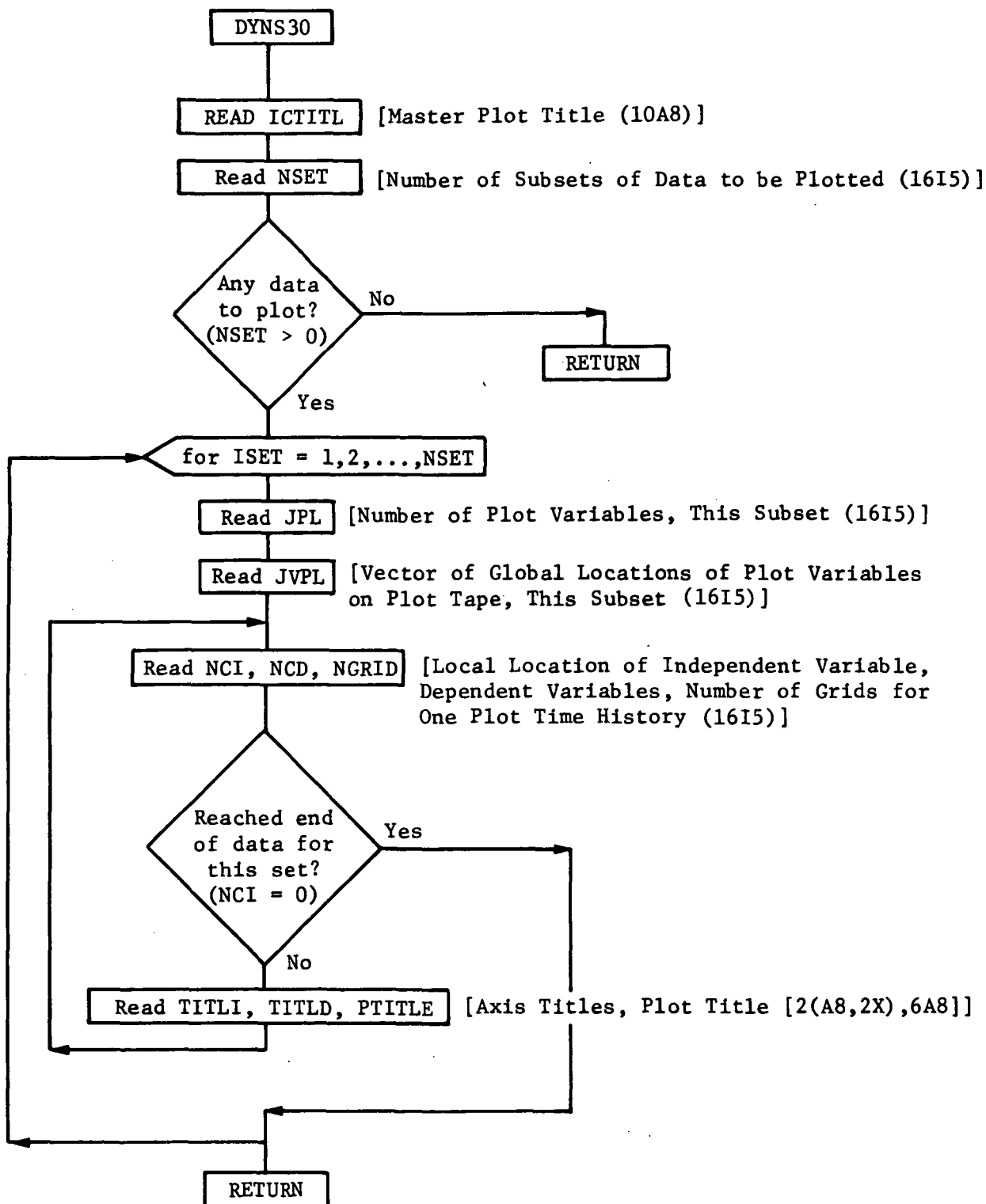


Figure 5. Program system DISCOS data stream flow (Sheet 10 of 12).

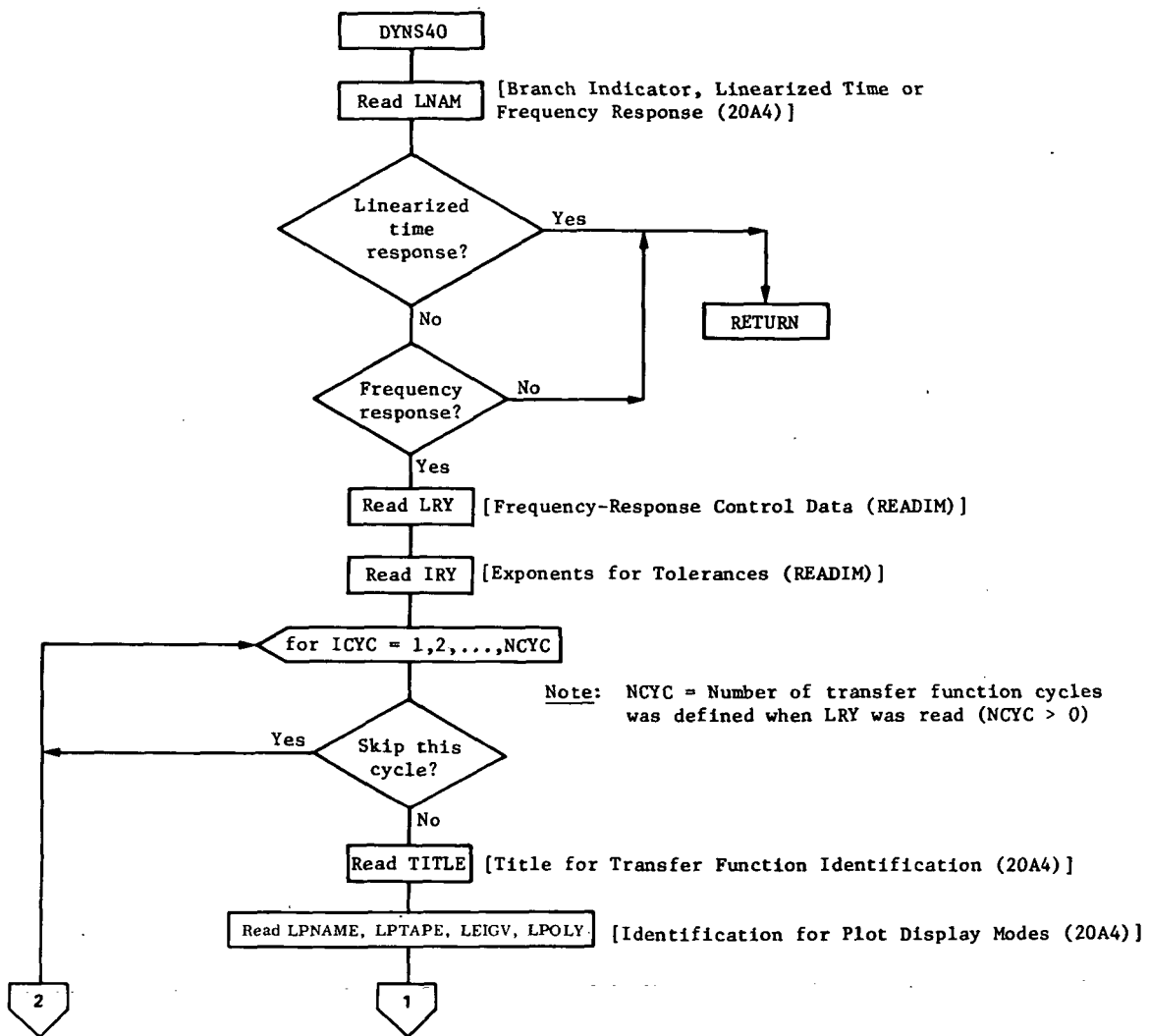


Figure 5. Program system DISCOS data stream flow (Sheet 11 of 12).

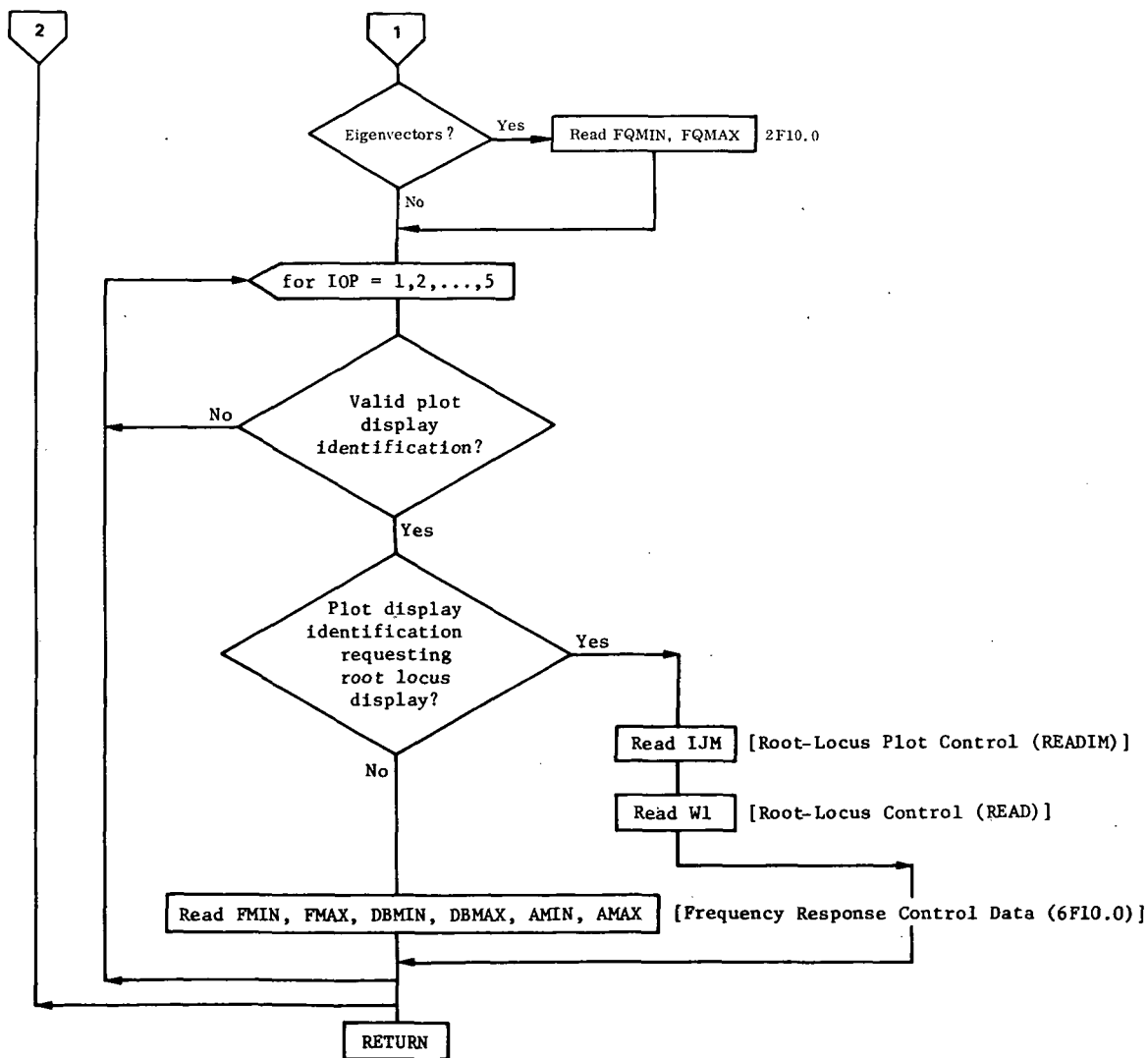


Figure 5. Program system DISCOS data stream flow (Sheet 12 of 12).

much the same manner as the input data stream of Section IV to better acquaint the reader with the actual formatted output as it is presented by the program.

Note that the output stream will not reflect certain outputs that occur from routines that identify troublesome areas such as matrix singularities. Recall also that the basic input routines, READ and READIM, can also print out input matrix data as dictated by the user. These printouts will not be included. Refer to the theoretical volume (Volume I) and to the input data stream (Section IV) for correlating certain outputs with both the theory and the user input requirements.

Again, DISCOS documentation is intended to grow as new capability is added. The user is referred to subroutines DEF4 and DEF5 for a detailed description of the standard DISCOS output. In addition to the standard output, if any or all of the "DEBUG" flags are set by subroutine START input data, additional output data are obtained. The setting of these DEBUG flags for initial checkout work is encouraged. The associated output will provide the user with a detailed definition of the computation performed internally and resultant parametric magnitudes. The comment cards in subroutine DEF5 and the associated subroutines provide detailed definitions of parameters outputted.

VI. AUXILIARY PROGRAMS

This section describes three auxiliary digital codes that have been developed to aid the DISCOS program system user. The first code is a FORTRAN program that accepts the DISCOS code as input and, based on some additional user-supplied input, automatically redimensions the DISCOS source program to minimize core storage requirements. It can also be used to update the source tape with new versions of old subroutines. The other two digital codes are used to obtain NASTRAN-generated finite-element data, which define body flexibility, directly from NASTRAN and to process it into a DISCOS-acceptable format.

A. REDIM—The Redimension Program

The REDIM code was developed to aid the user in the efficient use of available digital-computer core-storage locations. Examination of existing digital-computer codes for generalized analyses of (possibly) large systems indicates that the nature of the code frequently dictates that a great deal of core-storage locations are required because of the sizes of program DIMENSION and COMMON blocks. This often leads to inefficient use of core storage because the user must have available sufficient core-storage locations to satisfy the program size. As a large percentage of program executions probably do not require the maximum dimension sizes of program storage blocks, it is obvious that an automatic procedure for altering the program code to meet specific requirements of a user would be desirable. Program REDIM was developed to satisfy these requirements.

REDIM is a self-contained code that contains an extensive list of format statements. The code reads the DISCOS source code from tape as coded data and reproduces it on the tape

unless it finds an identifying format number in columns 73 through 75. In this case, it rewrites the source code according to the format corresponding to the identification. Therefore, REDIM provides an efficient and foolproof method of recasting the source input code to meet the user's requirements.

The following is a more detailed explanation of the manner in which the REDIM program can be used. Input data for the REDIM program are as follows:

- Data tapes:

NIT = 2 = (Input tape) the data-set reference number associated with the DISCOS source code to be redimensioned and/or updated

NOT = 1 = (Output tape) the data-set reference number associated with the DISCOS source code that has been redimensioned and/or updated

- Input Cards:

Card 1 READ (5,109) NCHNG
 109 FORMAT (I5)

NCHNG = Total number of subroutines on the input tape that will be totally updated

IF(NCHNG.EQ.O), this is the only data card required. The program immediately begins to read the source code from data set 2, to redimension it, and to output the redimensioned source on data set 1.

IF(NCHNG.GT.O), exactly NCHNG subroutines are to be updated. Because all subroutines are consecutively numbered by the DEBUG numbers, the subroutines to be updated are defined by the following input data cards:

READ(5,108) (NSUB(I),I=1),NCHNG)
108 FORMAT (16I5)

where

NSUB(I) = DEBUG number associated with the Ith-subroutine to be updated

All updated subroutines are read by the card reader immediately after the cards used to define the elements of NSUB. They are assumed to be in order of increasing DEBUG number. No separation cards are required between subroutines.

Subroutine DEF5 begins with a definition of the "maximum simulation model-size parameters." Each of these parameters can be altered in REDIM. The user is expected to simply change the appropriate variables in the REDIM source deck to accommodate specific redimensioning needs.

A listing of program REDIM follows:

```

CCC  PROGRAM TO DIMENSION DISCOS PROGRAM
CCCC
C    REVISED NOV. 20, 1974.
CCCC
C
      INTEGER IN(20)
      INTEGER NJN(120),NSUR(120)
      LOGICAL FLAG
      DIMENSION ICARD(18)
      DATA NIT, NOT / 2, 1 /
      DATA IEN, ID / 4H EN, 4HD /
C
      DATA NBMAX,NHMAX,NSPMAX,NMWMAX,NMWROD,NMDROD,KY,JMAXC,JMAXL /
*      6, 6, 15, 5, 4, 12,250, 7, 500 /
      DATA MAXZP,MXJVPL,MAXDUM /
*      1000, 16, 1500 /
      DATA MAXCNT /
*      150 /
      DATA L1,L5 / 99, 500 /
CCC
CCC  L1 MUST BE AN EVEN INTEGER.
CCC  IF NOT, IT WILL BE INCREMENTED BY 1
CCC  -----
CCC
C
1002 FORMAT (18A4,I3,I5)
708 FORMAT (1X,18A4,I3,I5)
703 FORMAT (1H1)
1  FORMAT (5X1H*5X4HAMU(I2,1H,I2,1H,I2,5H),8W(I2,1H,I3,1H)37X,I3,I5)
2  FORMAT (5X1H*5X5H8H(6,I2,1H,I2,7H),BS(6,I2,1H,I2,10H),ROL(3,3,I2,
* 8H),DOL(3,I2,1H)16X,I3,I5)
3  FORMAT (5X1H*5X4HGGG(I2,3H,9,I2,1H)49X,I3,I5)
4  FORMAT (5X1H*5X7HHATH(3,I2,1H,I2,9H),SIGH(3,I2,1H,I2,9H),HATS(3,
* I2,1H,I2,9H),SIGS(3,I2,1H,I2,1H)6X,I3,I5)
5  FORMAT (5X1H*5X3HAM(I3,1H,I2,9H),ACOF(9,I2,1H,I2,
* 9H),BCOF(6,I2,1H,I2,2H),22X,I3,I5)
6  FORMAT (5X1H*5X6HCOF11(I2,1H,I2,1H,I2,8H),COF22(I2,1H,I2,1H,I2,
* 8H),COF33(I2,1H,I2,1H,I2,5H),AK(I2,1H,I2,1H,I2,2H),I3,I5)
7  FORMAT (5X1H*5X6HCOF12(I2,1H,I2,1H,I2,8H),COF13(I2,1H,I2,1H,I2,
* 8H),COF23(I2,1H,I2,1H,I2,5H),AD(I2,1H,I2,1H,I2,2H),I3,I5)
8  FORMAT (5X1H*5X6HCOFXY(I2,1H,I2,1H,I2,8H),COFXZ(I2,1H,I2,1H,I2,
* 8H),COFYZ(I2,1H,I2,1H,I2,1H)14X,I3,I5)
9  FORMAT (5X1H*5X5HIV(6,I2,1H)53X,I3,I5)
10 FORMAT (5X1H*5X5HALAM(I2,1H)53X,I3,I5)
11 FORMAT (5X1H*5X2HP(I3,7H),PMOM(I2,24H),HTOT(3),TOTL(3),ENGKE(I2,
* 8H),ENGPE(I2,2H),9X,I3,I5)
12 FORMAT (5X1H*5X6HNNHPOI(I2,9H),NSPOI(I2,1H)41X,I3,I5)
13 FORMAT (5X1H*5X8HPIN(3,3,I2,11H),RP2(3,3,I2,11H),RP3(3,3,I2,
* 1H)24X,I3,I5)
14 FORMAT (5X1H*5X3HPR(I3,3H,5)52X,I3,I5)
15 FORMAT (5X1H*5X4HORK(I3,12H),PRK(4),NT42X,I3,I5)
16 FORMAT (5X1H*5X8HBETAH(6,I2,11H),BETAHD(6,I2,8H),AMO(2,I2,
* 9H),RH(3,3,I2,9H),RS(3,3,I2,2H),4X,I3,I5)
17 FORMAT (5X1H*5X5HDH(3,I2,7H),DS(3,I2,8H),IMO(3,I2,
* 7H),NMOW(I1,1H,I2,9H),IFTSMW(I2,2H),11X,I3,I5)
18 FORMAT (5X1H*5X3HNB,NH,NSPT,NOFMO,NDELTA,ITOPOL(2,I2,
* 9H),IRGFLX(I2,11H),IHDATA(7,I2,2H),I3,I5)
19 FORMAT (5X1H*5X5HLOCU(I2,7H),LENU(I2,11H),NU,NBETA,
* 8HNLAM,NEQ26X,I3,I5)

```

```

20 FORMAT (5X1H*5X2HY(I3,6H),YDT(I3,1H)46X,I3,I5)
21 FORMAT (5X1H*5X6HINDEP(I3,1H)51X,I3,I5)
41 FORMAT (6X9HNBMAX = I2,55X,I3,I5)
42 FORMAT (6X9HNNHMAX = I2,55X,I3,I5)
43 FORMAT (6X9HNSPMAX = I2,55X,I3,I5)
44 FORMAT (6X9HNMWMAX = I2,55X,I3,I5)
45 FORMAT (6X9HNMWBOD = I2,55X,I3,I5)
46 FORMAT (6X9HNMDBOD = I2,55X,I3,I5)
47 FORMAT (6X9HKMU = I2,55X,I3,I5)
48 FORMAT (6X9HKY = I3,54X,I3,I5)
49 FORMAT (6X9HKU = I3,54X,I3,I5)
61 FORMAT (6X12HDIMENSION A(I3,1H,I3,4H),B(I3,1H,I3,5H),IV(I3,
* 5H),JV(I3,19H),C(6,6),PHR6(6,6),1X,I3,I5)
62 FORMAT (5X1H*3X5HBC(9,I2,6H),WS1(I2,1H,I2,6H),WS2(I2,1H,I2,
* 6H),OM2(I3,8H),OMGA2(I2,7H),JDOF(I2,3H,6)3X,I3,I5)
63 FORMAT (6X28HDATA NIT,NOT,KAB,KJDOF/ 5,6,I3,1H,I2,2H /30X,I3,I5)
64 FORMAT (6X12HDIMENSION A(I4,1H,I2,4H),B(I4,1H,I2,
* 4H),C(I4,1H,I2,6H),AMU(I3,1H,I3,1H)11X,I3,I5)
65 FORMAT (6X13HDIMENSION WS(I4,1H,I2,7H),UVEC(I4,
* 7H),WV(3)28X,I3,I5)
66 FORMAT (5X5H* /5X,I3,1H,4X,I2,4H /43X,I3,I5)
69 FORMAT (6X35HDIMENSION A(KAR,1), R(KAR,1), IVEC(I3,1H)27X,I3,I5)
70 FORMAT (6X31HDIMENSION AZ(KAZ,1), B(KB,1),W(I3,1H)31X,I3,I5)
71 FORMAT (6X13HIF (NRB.GT. I3,11H) GO TO 99939X,I3,I5)
72 FORMAT (6X31HDIMENSION AZ(KAZ,1), B(KB,1),W(I3,1H)31X,I3,I5)
73 FORMAT (6X11HIF (NRB.GT.I3,13H .OR. NCB.GT.I3,
* 33H .OR. NRB.GT.KAZ .OR. NCB.GT.KAZ)3X,I3,I5)
74 FORMAT (6X15HDIMENSION ICON(I2,8H), IVEC(I2,7H),JCON(I3,
* 4H),R(I2,9H),JBIGST(I2,9H),JBIGFN(I2,1H)I3,I5)
75 FORMAT (6X14HDIMENSION GGV(I3,4H),D(I2,4H),V(I2,9H),BDTQ(6,I2,
* 9H),BDTP(6,I2,1H)14X,I3,I5)
76 FORMAT (6X14HDIMENSION BMR(I2,1H,I2,7H),BM(6,I2,1H,I2,1H)34X,I3,
* I5)
77 FORMAT (6X33HEQUIVALENCE (BW(1),BMR(1)), (BW(I4,8H),BM(1))21X,I3,
* I5)
78 FORMAT (6X13HDIMENSION FY(I3,6H,3),Z(I3,7H),ZNEW(I3,5H),IV(I3,
* 1H)22X,I3,I5)
79 FORMAT (6X30HEQUIVALENCE (PR(1),FY(1)),(PR(I4,12H),Z(1)),(PR(I4,
* 10H),ZNEW(1))6X,I3,I5)
80 FORMAT (6X29HDIMENSION A(KA,1),R(KA,1),CL(I3,1H)33X,I3,I5)
81 FORMAT (5X1H*8X8HITOPW(2,I2,8H),IOP(2,I2,8H),NBODF(I2,1H)27X,I3,
* I5)
82 FORMAT (6X14HDIMENSION BMR(I2,1H,I2,7H),BM(6,I2,1H,I2,
* 8H), ITOP(I2,1H,I2,5H),WR(I2,1H)14X,I3,I5)
83 FORMAT (6X15HDIMENSION RW(3,I2,5H),CW(I2,
* 15H,3),VW(9),WV(6)27X,I3,I5)
84 FORMAT (6X40HDIMENSION A(KRA,1),B(KRH,1),Z(KRZ,1),WR(I3,
* 1H)22X,I3,I5)
85 FORMAT (6X40HDIMENSION A(KRA,1),B(KRH,1),Z(KRZ,1),WR(I3,
* 1H)22X,I3,I5)
86 FORMAT (6X44HDIMENSION A(6,1),B(KMU,1),C(6,1),IV(6,1),RW(I2,
* 1H)19X,I3,I5)
87 FORMAT (6X13HDIMENSION CW(I2,9H,3),RW(3,I2,5H),V1(I2,
* 17H),WSK(3,3),TSHFT(I2,2H),12X,I3,I5)
88 FORMAT (5X1H*5X6HTEX(6,I2,7H),ISPN(I2,10H),V(6),V2(I2,1H)
* 31X,I3,I5)
89 FORMAT (6X13HDIMENSION WV(I2,1H)50X,I3,I5)
90 FORMAT (6X13HDIMENSION ZP(I4,1H,I2,6H),DUM(I4,7H),JVPL(I2,
* 1H)26X,I3,I5)
91 FORMAT (6X20HDATA KRPLT,KCPLT /I4,1H,I2,1H/38X,I3,I5)
92 FORMAT (6X17HDIMENSION HNGT(6,I2,1H)46X,I3,I5)

```

```

93 FORMAT (6X16HDIMENSION VCM(3,12,10H),TIN(3,3,12,27H),SYSCM(3),SYSI
  *N(3,3),XMAS(12,1H)6X,I3,I5)
95 FORMAT (5X1H*5X7HCNTDTA(I3,1H)50XI3,I5)
96 FORMAT (6X8HKCONT = I3,55XI3,I5)
97 FORMAT (6X13HDIMENSION VI(14,5H),VD(14,1H)39XI3,I5)
98 FORMAT (5X1H*5X7HASUMRY(I2,11H,6),ISUMRY(I2,
  * 10H,3),KSUMRY29X,I3,I5)
99 FORMAT (6X9HKSUMRY = I2,55X,I3,I5)
100 FORMAT (5X1H*5X16HIFL1,IFL2,DRVEC(I3,1H)41X,I3,I5)

C
410 FORMAT (6X,12HDIMENSION Q(I3,1H,,I3,6H), V1(I3,1H),37X,I3,I5)
411 FORMAT (6X,29HEQUIVALENCE (Q(1),W1(1)), (Q(,14,8H),W2(1)),25X,
  * I3,I5)
412 FORMAT (7X14HDIMENSION VS1(I3,6H),VS2(I3,1H)38X,I3,I5) 412
414 FORMAT (7X,14HDIMENSION RRN(I3,6H),RIN(I3,6H),RRD(I3,6H),RID(
  * I3,6H),R2R(I3,6H),R2I(I3,1H)2XI3,I5) 414
415 FORMAT (6X,17HDIMENSION ROOTRE(I3,9H),ROOTIE(I3,4H),W(I3,
  * 9H,4),IROW(I3,3H,2),12X,I3,I5)
416 FORMAT (6X,13HDIMENSION XR(I3,5H),XI(I3,5H),VR(I3,5H),VI(I3,1H),
  * 25X,I3,I5)
418 FORMAT (7X16HDIMENSION IJM(2,I3,8H),LRY(9,I3,
  * 9H), IRY(3,I3,10H), KBKP(3),13X,I3,I5) 418
420 FORMAT (5X1H915X3HIV(I3,10H) , JV (I3,1H),31X,I3,I5) 420
422 FORMAT (5X1HA15X5HFBRN(I3,8H), FBNC(I3,8H), FBRD(I3,8H), FBDC(
  * I3,1H)9XI3,I5) 422
424 FORMAT (5X1HB15X2HR(I3,11H) , RX (I3,1H)31XI3,I5) 424
426 FORMAT (5X1HC15X5HV1 (I3,8H), V2 (I3,8H), V3 (I3,1H)20XI3,I5) 426
428 FORMAT (5X1HD15X5HXV1 (I3,8H), XV2 (I3,8H), XV3 (I3,
  * 8H), XV4 (I3,1H)9XI3,I5) 428
430 FORMAT (5X1HE15X5HY (I3,8H), YD (I3,1H)31XI3,I5) 430
432 FORMAT (5X1HF15X3HW1(I3,1H,I3,6H), W2(I3,1H,I3,1H)27XI3,I5) 432
434 FORMAT (6X17HEQUIVALENCE (VS1(I3,9H),VS2(1))37XI3,I5) 434
436 FORMAT (6X6HKKR = I3,57XI3,I5) 436
438 FORMAT (6X6HKKRT = I3,57XI3,I5) 438
440 FORMAT (6X6HKKRX = I3,57XI3,I5) 440
442 FORMAT (6X6HKV1 = I3,57XI3,I5) 442
444 FORMAT (6X6HKV2 = I3,57XI3,I5) 444
446 FORMAT (6X6HKVX = I3,57XI3,I5) 446
447 FORMAT (5X1H*12X6HSAVED(I3,9H), SAVEP(I3,9H), SAVED(I3,
  * 9H), SAVEA(I3,8H), KSAVE ,1XI3,I5) 447
448 FORMAT (6X15HDIMENSION IVEC(I3,8H), JVEC(I3,1H)36XI3,I5)
450 FORMAT (6X12HDIMENSION A(I3,4H),R(I3,4H),W(I3,4H),B(I3,4H),C(I3,
  * 5H),SS(I3,1H)14XI3,I5) 450
452 FORMAT (6X14HDIMENSION PAR(I3,7H), QAR(I3,1H)38XI3,I5) 452
454 FORMAT (6X16HDIMENSION XSAVE(I3,8H),YSAVE(I3,1H)35XI3,I5) 454
456 FORMAT (5X12H1 6 , I3,4H /,48XI3,I5) 456
460 FORMAT (6X14HDIMENSION PAR(I3,7H), QAR(I3,1H)38XI3,I5) 460
462 FORMAT (5X1H110X15H TITLE(1) , WD(I3,
  1 25H) , TARG(20) , TARUP(31),13XI3,I5)
464 FORMAT (5X1H310X5HENUM(I3,8H), EDEN(I3,1H)36XI3,I5) 464
466 FORMAT (6X9HDATA KWD/I3,1H/,53XI3,I5)
468 FORMAT (6X12HDATA KDSAVE/I3,1H/50XI3,I5)
470 FORMAT (6X22HDIMENSION PRK(4), YDS(I3,8H,3), YS(I3,1H)29XI3,I5) 470
472 FORMAT (6X22HDIMENSION TITLE(1), X(I3,5H), Y(I3,1H)32XI3,I5) 472
474 FORMAT (6X15HDIMENSION FAZE(I3,1H)7H,DBAMP(I3,1H)36XI3,I5) 474
480 FORMAT (6X18HCOMMON /LWRKV1/ W(I3,1H)44XI3,I5)

```

```

IGASMX = 150
MLTAMX = 150
MXBTAR = 150

```

```

MLT3MX = 100
MADMAX = 100
C
  IF (MOD(L1,2) .EQ. 1) L1 = L1 + 1
  L2 = 2*L1 + 7
  L3 = L1*L1/2 + 1
  L4 = 2*L2
  L2P1 = L2 + 1
  L01 = NBMAX
  L02 = NHMAX
  L03 = NSPMAX
  L04 = NMWMAX
  L05 = NMWBOD
  L06 = NMDBOD
  L07 = KY
  L08 = 6 + NMDBOD + NMWBOD
  L09 = 6*NHMAX
  L10 = NBMAX*(6 + NMDBOD) + NMWMAX
  L11 = 6 + NMDBOD
  L12 = 2*NHMAX - 1
  L13 = 2*(NHMAX - 1)
  L14 = 6*NBMAX
  L15 = 6*(NHMAX - 1)
  L16 = 7*(NHMAX - 1)
  L17 = 2*NSPMAX
  L18 = 2 + NMWBOD
  L19 = 2*NBMAX + 2
C
  L20 = L09*L09 + 1
  L40 = 25
  IF (L06 .GT. 12) L40 = 2*L06+1
  L41 = JMAXC
  L42 = 6*JMAXC
  L43 = JMAXL
  L44 = 13
  L45 = NMDBOD
  L46 = IGASMX
  L47 = MLTAMX
  L48 = MXBTAB
  L49 = 3*L07+1
  L50 = 4*L07+1
  L51 = MLT3MX
  L52 = MADMAX
  L53 = MAXZP
  L54 = MXJVPL
  L55 = MAXDUM
  L56 = MAXCNT
  L57 = MAX0(9,NMDBOD)
  L58 = MAX0(10,NMDBOD)
  L59 = ((L06+6)*(L06+7))/2
  L60 = MAX0(L02,L03)
C
  KC1 = L09*L10
  KC2 = L09*L09+6*L08*L12
  IF (KC2 .GT. KC1) L10 = KC2/L09 + 1
C
CCCCCCCC
  REWIND NIT
CCCCCCCC
  DO 805 I=1,120
    805 NIN(I) = 2
      READ(5,109) NCHNG
      WRITE(6,209) NCHNG

```



```

209 FORMAT (15,' UPDATED SUBROUTINES WILL BE READ IN')
      IF(NCHNG.EQ.0) GO TO 812
      READ(5,108) (NSUB(I),I=1,NCHNG)
      WRITE(6,208) (NSUB(I),I=1,NCHNG)
208 FORMAT (' DEBUG NUMBERS FOR THE SUBROUTINES ARE:',16I5)
      DO 806 I=1,NCHNG
      NN = NSUB(I)
806 NIN(NN) = 5
812 CONTINUE
      ICNT = 1
      M = 0
500 CONTINUE
      M = M + 1
      NNN = NIN(ICNT)
      READ (NNN,1002,END=700) ICARD,N,MM
      IF(ICARD(2).EQ.1EN.AND.ICARD(3).EQ.ID) GO TO 810
      GO TO 808
810 IF(NIN(ICNT).EQ.5) WRITE(NOT,1002) ICARD,N,M
807 IF(NIN(ICNT).EQ.2) GO TO 809
      READ(2,1002,END=700) ICARD,N,MM
      IF(ICARD(2).EQ.1EN.AND.ICARD(3).EQ.ID) GO TO 811
      GO TO 807
811 ICNT = ICNT + 1
      GO TO 500
809 ICNT = ICNT + 1
C
808 CONTINUE
      IF (N .GT. 0) GO TO 600
      WRITE (NOT,1002) ICARD,N,M
      GO TO 500
600 IF (N .LT. 11) GO TO 601
      IF (N .LT. 21) GO TO 602
      IF (N .LT. 49) GO TO 603
      IF (N .LT. 70) GO TO 604
      IF (N .LT. 80) GO TO 605
      IF (N .LT.101) GO TO 606
      IF (N .LT. 428) GO TO 610
      IF (N .LT. 450) GO TO 612
      GO TO 615
601 IF(N.EQ. 1)WRITE (NOT,1)  L08,L08,L01,L09,L10,N,M
      IF(N.EQ. 2)WRITE (NOT,2)  L11,L12,L11,L03,L01,L01,N,M
      IF(N.EQ. 3)WRITE (NOT,3)  L06,L01,N,M
      IF(N.EQ. 4)WRITE (NOT,4)  L06,L13,L06,L13,L06,L03,L06,L03,N,M
      IF(N.EQ. 5)WRITE (NOT,5)  L59,L01,L06,L01,L06,L01,N,M
      IF(N.EQ. 6)WRITE (NOT,6)  L06,L06,L01,L06,L06,L01,L06,L06,L01,
*      L06,L06,L01,N,M
      IF(N.EQ. 7)WRITE (NOT,7)  L06,L06,L01,L06,L06,L01,L06,L06,L01,
*      L06,L06,L01,N,M
      IF(N.EQ. 8)WRITE (NOT,8)  L06,L06,L01,L06,L06,L01,L06,L06,L01,N,M
      IF(N.EQ. 9)WRITE (NOT,9)  L02,N,M
      IF(N.EQ.10)WRITE (NOT,10) L09,N,M
      GO TO 500
602 IF(N.EQ.11)WRITE (NOT,11) L10,L14,L01,L01,N,M
      IF(N.EQ.12)WRITE (NOT,12) L01,L01,N,M
      IF(N.EQ.13)WRITE (NOT,13) L02,L02,L02,N,M
      IF(N.EQ.14)WRITE (NOT,14) L07,N,M
      IF(N.EQ.15)WRITE (NOT,15) L07,N,M
      IF(N.EQ.16)WRITE (NOT,16) L02,L02,L04,L15,L17,N,M
      IF(N.EQ.17)WRITE (NOT,17) L16,L17,L04,L18,L01,L03,N,M
      IF(N.EQ.18)WRITE (NOT,18) L02,L01,L02,N,M
      IF(N.EQ.19)WRITE (NOT,19) L19,L19,N,M
      IF(N.EQ.20)WRITE (NOT,20) L07,L07,N,M
      GO TO 500

```

```

603 IF(N.EQ.21)WRITE (NOT,21) L07,N,M
    IF(N.EQ.41)WRITE (NOT,41) L01,N,M
    IF(N.EQ.42)WRITE (NOT,42) L02,N,M
    IF(N.EQ.43)WRITE (NOT,43) L03,N,M
    IF(N.EQ.44)WRITE (NOT,44) L04,N,M
    IF(N.EQ.45)WRITE (NOT,45) L05,N,M
    IF(N.EQ.46)WRITE (NOT,46) L06,N,M
    IF(N.EQ.47)WRITE (NOT,47) L08,N,M
    IF(N.EQ.48)WRITE (NOT,48) L07,N,M
    GO TO 500
604 IF(N.EQ.49)WRITE (NOT,49) L10,N,M
    IF(N.EQ.61)WRITE (NOT,61) L42,L42,L42,L42,L42,L42,N,M
    IF(N.EQ.62)WRITE (NOT,62) L06,L06,L06,L06,L06,L42,L11,L41,N,M
    IF(N.EQ.63)WRITE (NOT,63) L42,L41,N,M
    IF(N.EQ.64)WRITE(NOT,64) L43,L45,L43,L45,L43,L45,L11,L11,N,M
    IF(N.EQ.65)WRITE(NOT,65) L43,L44,L43,N,M
    IF(N.EQ.66)WRITE(NOT,66) L43,L45,N,M
    IF(N.EQ.69)WRITE (NOT,69) L46,N,M
    GO TO 500
605 IF(N.EQ.70)WRITE (NOT,70) L47,N,M
    IF(N.EQ.71)WRITE (NOT,71) L47,N,M
    IF(N.EQ.72)WRITE (NOT,72) L48,N,M
    IF(N.EQ.73)WRITE (NOT,73) L48,L48,N,M
    IF(N.EQ.74)WRITE (NOT,74) L09,L09,L10,L09,L02,L02,N,M
    IF(N.EQ.75)WRITE (NOT,75) L10,L09,L09,L11,L11,N,M
    IF(N.EQ.76)WRITE (NOT,76) L09,L09,L08,L12,N,M
    IF(N.EQ.77)WRITE (NOT,77) L20,N,M
    IF(N.EQ.78)WRITE (NOT,78) L07,L07,L07,L07,N,M
    IF(N.EQ.79)WRITE (NOT,79) L49,L50,N,M
    GO TO 500
606 IF(N.EQ.80)WRITE (NOT,80) L08,N,M
    IF(N.EQ.81)WRITE (NOT,81) L02,L02,L01,N,M
    IF(N.EQ.82)WRITE (NOT,82) L09,L09,L08,L12,L02,L01,L11,N,M
    IF(N.EQ.83)WRITE (NOT,83) L06,L06,N,M
    IF(N.EQ.84)WRITE (NOT,84) L51,N,M
    IF(N.EQ.85)WRITE (NOT,85) L52,N,M
    IF(N.EQ.86)WRITE (NOT,86) L11,N,M
    IF(N.EQ.87)WRITE (NOT,87) L06,L06,L06,L04,N,M
    IF(N.EQ.88)WRITE (NOT,88) L03,L03,L06,N,M
    IF(N.EQ.89)WRITE (NOT,89) L06,N,M
    IF(N.EQ.90)WRITE (NOT,90) L53,L54, L55,L54,N,M
    IF(N.EQ.91)WRITE (NOT,91) L53,L54,N,M
    IF(N.EQ.92)WRITE (NOT,92) L02,N,M
    IF(N.EQ.93)WRITE(NOT,93) L01,L01,L01,N,M
    IF(N.EQ.95)WRITE (NOT,95) L56,N,M
    IF(N.EQ.96)WRITE (NOT,96) L56,N,M
    IF(N.EQ.97)WRITE (NOT,97) L53,L53,N,M
    IF(N.EQ.98)WRITE(NOT,98) L60,L60,N,M
    IF(N.EQ.99)WRITE(NOT,99) L60,N,M
    IF (N.EQ.100)WRITE(NOT,100) L46,N,M
    GO TO 500
610 CONTINUE
    IF (N .EQ. 410) WRITE (NOT,410) L1,L1,L1,N,M
    IF (N .EQ. 411) WRITE (NOT,411) L3,N,M
    IF (N .EQ. 412) WRITE(NOT,412) L4,L2,N,M
    IF (N .EQ. 414) WRITE (NOT,414) (L1,I=1,6),N,M
    IF(N.EQ.415)WRITE (NOT,415) L1,L1,L1,L1,N,M
    IF(N.EQ.416)WRITE (NOT,416) L1,L1,L1,L1,N,M
    IF (N .EQ. 418) WRITE (NOT,418) L1,L1,L1,N,M
    IF (N .EQ. 420) WRITE (NOT,420) L07,L07,N,M
    IF (N .EQ. 422) WRITE (NOT,422) (L1,I=1,4),N,M
    IF (N .EQ. 424) WRITE (NOT,424) L2,L4,N,M

```

```

        IF (N .EQ. 426) WRITE (NOT,426) L1,L1,L1,N,M
        GO TO 500
612 CONTINUE
        IF (N .EQ. 428) WRITE (NOT,428) (L1,I=1,4),N,M
        IF (N .EQ. 430) WRITE (NOT,430) L07,L07,N,M
        IF (N .EQ. 432) WRITE (NOT,432) (L1,I=1,4),N,M
        IF (N .EQ. 434) WRITE (NOT,434) L2P1,N,M
        IF (N .EQ. 436) WRITE (NOT,436) L1,N,M
        IF (N .EQ. 438) WRITE (NOT,438) L2,N,M
        IF (N .EQ. 440) WRITE (NOT,440) L4,N,M
        IF (N .EQ. 442) WRITE (NOT,442) L4,N,M
        IF (N .EQ. 444) WRITE (NOT,444) L2,N,M
        IF (N .EQ. 446) WRITE (NOT,446) L1,N,M
        IF (N .EQ. 447) WRITE (NOT,447) (L5,I=1,4),N,M
        IF (N .EQ. 448) WRITE (NOT,448) L2,L2,N,M
        GO TO 500
615 CONTINUE
        IF (N .EQ. 450) WRITE (NOT,450) (L1,I=1,6),N,M
        IF (N .EQ. 452) WRITE (NOT,452) L1, L1, N, M
        IF (N .EQ. 454) WRITE (NOT,454) L5, L5, N, M
        IF (N .EQ. 456) WRITE (NOT,456) L5,N,M
        IF (N .EQ. 460) WRITE (NOT,460) L1,L1,N,M
        IF (N .EQ. 462) WRITE (NOT,462) L2,N,M
        IF (N .EQ. 464) WRITE (NOT,464) L1,L1,N,M
        IF (N .EQ. 466) WRITE (NOT,466) L2,N,M
        IF (N .EQ. 468) WRITE (NOT,468) L5,N,M
        IF (N .EQ. 470) WRITE (NOT,470) L07,L07,N,M
        IF (N .EQ. 472) WRITE (NOT,472) L5, L5, N, M
        IF (N .EQ. 474) WRITE (NOT,474) L5,L5,N,M
        IF (N .EQ. 480) WRITE (NOT,480) L1,N,M
        GO TO 500

```

C

```

700 REWIND 1
    IIC = 1
    WRITE(6,704) IIC
    IIC = 2
704 FORMAT (////,90X,' SUBROUTINE',I5)
706 READ (1,1002,END=707) ICARD,N,M
    WRITE (6,708) ICARD,N,M
    IF (ICARD(2).EQ.IEN .AND. ICARD(3).EQ.ID) GO TO 709
    GO TO 706
709 WRITE(6,704) IIC
    IIC = IIC+1
    GO TO 706
707 CONTINUE
    REWIND 1
    STOP
101 FORMAT(20A4)
102 FORMAT(18A4,18)
103 FORMAT(1X,A4,68X,18)
104 FORMAT(1X,18A4,18)
105 FORMAT('1')
106 FORMAT(A4,68X,18)
107 FORMAT (80X,12A4)
108 FORMAT(16I5)
109 FORMAT(15)
    END

```

B. Support Programs for the DISCOS Flexible-Body Capability

The flexible-body option available in the DISCOS program requires that the user obtain mode-dependent parameters. For most applications, these parameters will be computed from lumped parameter data obtained by the standard in-house finite-element program. Although NASTRAN is the standard Goddard Space Flight Center (GSFC), National Aeronautics and Space Administration (NASA), program, other installations make use of other programs. Therefore, the following two programs, submitted as a guide for the interested user, can either be used directly or be extensively modified to suit the situation at the particular computer installation at which DISCOS will be applied.

The first program is actually the NASTRAN Executive Control Deck for a specially written DMAP program. This DMAP program will output all lumped parameter data that are necessary for generating the required DISCOS input data. The N-BOD2 program referred to in the DMAP listing is another flexible-body program used at GSFC. It requires similar data but in a completely different format.

This DMAP program is operational on the MacNeal-Schwendler version 38 of NASTRAN and the Level 15.5 NASA version with one small change in the functional module, GP4. (Built-in NASTRAN error messages will point out the change.)

The DMAP program was written so that, in addition to modes and frequencies (real), NASTRAN could be forced to output modal mass, stiffness, and damping matrices. Several print and punch options were also built in so that the user could see more clearly some of the resultant mass stiffness and damping matrices internally generated by NASTRAN. For a detailed description of the DMAP program, the interested user is referred to a NASTRAN Level 15 User's Manual* and Programmer's Manual.† It is basically a combination of rigid formats 3, 9, and 12.

```
ID      FRISCH, DISCOS N-BOD2 NASTRAN
DIAG    14
TIME    6000
APP     DMAP
BEGIN   DISCOS AND N-BOD2 DATA VIA NASTRAN $
FILE    $
GP1     GEOM1,GEOM2,/GPL,EQEXIN,GPDT,CSTM,BGPDT,SIL/V,N,LUSET/ C,N,123/
        V,N,NOGPDT $
SAVE    LUSET,NOGPDT $
CHKPNT  GPL,EQEXIN,GPDT,CSTM,BGPDT,SIL $
PARAM   //C,N,NOP/V,Y,PRTGD=1 $
COND    LBL10,PRTGD $
TABPT   GPL,EQEXIN,GPDT,BGPDT,SIL// $
TABPRT  GPL//C,N,GPL $
TABPRT  GPDT//C,N,GPDT $
TABPRT  BGPDT//C,N,BGPDT $
LABEL   LBL10 $
```

*Calab W. McCormick, *The NASTRAN User's Manual*, NASA SP-222, September 1970.

†Frank J. Douglas, *The NASTRAN Programmer's Manual*, NASA SP-223, September 1970.

```

PURGE  USET,GM,GO,KAA,BAA,MAA,K4AA,PST,KFS,QP,EST/NOGPDT $
CHKPNT USET,GM,GO,KAA,BAA,MAA,K4AA,PST,KFS,QP,EST $
GP2    GEOM2,EQEXIN/ECT $
CHKPNT ECT $
TA1,   ,ECT,EPT,RGPD,T,SIL,,CSTM/EST,,GEI,ECPT,GPCT/V,N,LUSET/ C,N,
123/V,N,NOSIMP/C,N,O/V,N,NOGENL/V,N,GENEL $
SAVE   NOGENL,NOSIMP,GENEL $
CHKPNT EST,GEI,ECPT,GPCT $
PURGE  K4GG,GPST,OGPST,MGG,BGG, K4NN,K4FF,K4AA,MNN,MFF,MAA,BNN,BFF,
BAA,KGGX/NOSIMP/ OGPST/GENEL $
CHKPNT K4GG,GPST,OGPST,MGG,BGG,K4NN,K4FF,K4AA,MNN,MFF,MAA,BNN,BFF,
BAA,KGGX $
SMA1   CSTM,MPT,ECPT,GPCT, DIT/KGGX,K4GG,GPST/V,N,NOGENL/ V,N,NOK4GG$
SAVE   NOK4GG $
CHKPNT KGGX,K4GG,GPST $
PARAM  //C,N,NOP/V,Y,PRTKG=-1 $
COND   LBL14,PRTKG $
MATPRN KGGX,,,// $
LABEL  LBL14 $
PURGE  K4NN,K4GG,K4FF,K4AA/NOK4GG $
CHKPNT K4NN,K4GG,K4FF,K4AA $
SMA2   CSTM,MPT,ECPT,GPCT,DIT/MGG,BGG/V,Y,WTMASS=1.0/V,N,NOMGG/ V,N,
NOBGG/V,Y,COUPMASS=-1 $
SAVE   NOMGG,NOBGG $
CHKPNT MGG,BGG $
PARAM  //C,N,NOP/V,Y,PRTMG=-1 $
COND   LBL11,PRTMG $
MATPRN MGG,,,// $
LABEL  LBL11 $
PARAM  //C,N,NOP/V,Y,PRTMGD=1 $
COND   LBL15,PRTMGD $
DIAGONAL MGG/MMG/C,Y,OPT=COLUMN/V,Y,POWER=1. $
MATPRN MMG,,,// $
LABEL  LBL15 $
PURGE  BNN,BFF,BAA,BGG/NOBGG $
CHKPNT BNN,BFF,BAA,BGG $
COND   LBL1,GRDPNT $
GPWG   BGPDT,CSTM,EQEXIN,MGG/OGPWG/V,Y,GRDPNT=-1/V,Y,WTMASS $
CHKPNT OGPWG $
OFF    OGPWG,,,//V,N,CARDNO $
SAVE   CARDNO $
LABEL  LBL1 $
PARAM  //C,N,MPY/V,N,NSKIP/C,N,O/C,N,O $
GP4    CASECC,GEOM4,EQEXIN,SIL,GPDT,BGPDT,CSTM/RG,,USET,/V,N,LUSET/
V,N,MPCF1/V,N,MPCF2/V,N,SINGLE/V,N,OMIT/V,N,REACT/V,N,NSKIP/
V,N,REPEAT/V,N,NOSET/V,N,NOL/V,N,NOA/C,N,O $
SAVE   MPCF1,SINGLE,OMIT,NOSET,REACT,MPCF2,NSKIP,REPEAT,NOL,NOA $
CHKPNT RG,USET $
PURGE  GM,GMD/MPCF1/GO,KOOR,LOO,UOO,MOOB,MOAB,GOD/OMIT/KFS,PST,QP/
SINGLE $
CHKPNT GM,GMD,GO,KOOR,LOO,UOO,MOOB,MOAB,GOD,KFS,PST,QP $
EQUIV  KGGX,KNN/MPCF1/MGG,MNN/MPCF1/ BGG,BNN/MPCF1/K4GG,K4NN/MPCF1 $
CHKPNT KNN,MNN,BNN,K4NN,OGPST $
GPSP   GPL,GPST,USET,SIL/OGPST $
OFF    OGPST,,,//V,N,CARDNO $
SAVE   CARDNO $
COND   LBL2,MPCF1 $
MCE1   USET,RG/GM $
MCE2   USET,GM,KGGX,MGG,BGG,K4GG/KNN,MNN,BNN,K4NN $

```

```

CHKPNT GM,KNN,MNN,BNN,K4NN $
LABEL LBL2 $
EQUIV KNN,KFF/SINGLE/MNN,MFF/SINGLE/BNN,BFF/SINGLE/K4NN,K4FF/SINGLE $
CHKPNT KFF,MFF,BFF,K4FF $
COND LBL3,SINGLE $
SCE1 USET,KNN,MNN,BNN,K4NN/KFF,KFS, ,MFF,BFF,K4FF $
LABEL LBL3 $
EQUIV KFF,KAA/OMIT/ MFF,MAA/OMIT/BFF,BAA/OMIT/K4FF,K4AA/OMIT $
CHKPNT KAA,MAA,BAA,K4AA $
COND LBL20,OMIT $
SMP1 USET,KFF, ,BFF,K4FF/GO,KAA,KOGB,LOO,UOO, , ,BAA,K4AA $
CHKPNT GO,KAA,KOGB,LOO,UOO,BAA,K4AA $
SMP2 USET,GO,MFF/MAA $
CHKPNT MAA $
LABEL LBL20 $
PURGE KRR,KLR,DM,MLR,MR/REACT/CM/MPCF1/GO/OMIT/KFS/SINGLE/QG/NOSET $
COND LBL21,REACT $
RBMG1 USET,KAA,MAA/KLL,KLR,KRR,MLL,MLR,MRR $
CHKPNT KLL,KLR,KRR,MLL,MLR,MRR $
RBMG2 KLL/LLL,ULL $
CHKPNT ULL,LLL $
RBMG3 LLL,ULL,KLR,KRR/DM $
CHKPNT DM $
RBMG4 DM,MLL,MLR,MRR/MR $
CHKPNT MR $
LABEL LBL21 $
DPD DYNAMICS,GPL,SIL,USET/GPLD,SILD,USETD, , , , ,EED,EQDYN/V,N,
LUSET/V,N,LUSETD/V,N,NOTFL/V,N,NODLT/V,N,NOPSDL/V,N,NOFRL/V,N,
NOMLFT/V,N,NOTRL/V,N,NOEED/C,N,123/V,N,NOUE $
SAVE NOEED,NOUE $
CHKPNT GPLD,SILD,USETD,EED,EQDYN $
COND ERRO2,NOEED
READ KAA,MAA,MR,DM,EED,USET,CASECC/LAMA,PHIA,MI,OEIGS/C,N,MODES/ V,
N,NEIGV $
SAVE NEIGV$
CHKPNT LAMA,PHIA,MI,OEIGS $
ADD MI,/OI/C,N,(1.-10,0.0) $
OFP LAMA,OEIGS, , , ,/V,N,CARDNO $
COND FINIS,NEIGV $
SDR1 USET, ,PHIA, ,GO,GM, ,KFS, ,/PHIG, ,QG/C,N,1/C,N,REIG $
CHKPNT PHIG,QG $
PARAM //C,N,NOP/V,Y,PRTMD=1 $
COND LBL12,PRTMD $
MATPRN PHIG, , , ,/ $
LABEL LBL12 $
SDR2 CASECC,CSTM,MPT,DIT,EQEXIN,SIL, , ,BGPDT,LAMA,QG,PHIG,EST,/,OQG1,
OPHIG,OES1,DEF1,PPHIG/C,N,REIG $
CHKPNT OQG1,OPHIG,OES1,DEF1,PPHIG $
OFP OPHIG,OQG1,DEF1,OES1, ,/V,N,CARDNO $
SAVE CARDNO $
EQUIV GO,GOD/NOUE/GM,GMD/NOUE $
CHKPNT GOD,GMD $
MTRXIN CASECC,MATPOOL,EQDYN, ,/K2PP,M2PP,B2PP/V,N,LUSETD/V,N,NOK2PP/V,
N,NOM2PP/V,N,NOB2PP $
SAVE NOK2PP,NOM2PP,NOB2PP $
CHKPNT K2PP,M2PP,B2PP $
PARAM //C,N,AND/V,N,KDEKA/V,N,NOUE/V,N,NOK2PP $
PARAM //C,N,AND/V,N,MDEMA/V,N,NOUE/V,N,NOM2PP $
PARAM //C,N,AND/V,N,KDEK2/V,N,NOGENL/V,N,NOSIMP $

```

```

PURGE K2DD/NOK2PP/M2DD/NOM2PP/R2DD/NOB2PP $
EQUIV M2PP,M2DD/NOA/B2PP,B2DD/NOA/K2PP,K2DD/NOA/MAA,MDD/MDEMA/ KAA,
      KDD/KDEKA
GKAD  USETD,GM,GO,KAA,BAA,MAA,K4AA,K2PP,M2PP,B2PP/KDD,BDD,MDD,GMD,
      GOD,K2DD,M2DD,B2DD/C,N,TRANRESP/C,N,DISP/C,N,DIRECT/C,Y,G=0.0/
      C,Y,W3=0.0/C,Y,W4=0.0/V,N,NOK2PP/V,N,NOM2PP/V,N,NOB2PP/ V,N,
      MPCF1/V,N,SINGLE/V,N,OMIT/V,N,NOUE/V,N,NOK4GG/V,N,NORGG/ V,N,
      KDEK2/C,N,-1 $
CHKPNT M2DD,B2DD,K2DD,MDD,KDD,BDD,GMD,GOD $
GKAM  USETD,PHIA,OI,LAMA,DIT,MDD,BDD,KDD,CASECC/MHH,BHH,KHH,
      PHIDH/V,N,NOUE/C,Y,LMODES=0/C,Y,LFREQ=0.0/ C,Y,HFREQ=0.0/C,N,
      1/C,N,1/C,N,1/V,N,NONCPU/V,N,FMODE $
CHKPNT MHH,BHH,KHH,PHIDH $
PARAM //C,N,NOP/V,Y,PRMM=1 $
COND  LBL13,PRMM $
MATPRN KHH,BHH,MHH,PHIDH,//$ $
LABEL LBL13 $*
PARAM //C,N,NOP/V,Y,PCHGD=-1 $
COND  LBL6,PCHGD $
TABPCH GPL,BGPD,.,.,//C,N,GP/C,N,BG $
LABEL LBL6 $
PARAM //C,N,NOP/V,Y,PCHMG=-1 $
COND  LBL7,PCHMG $
OUTPUT3 MGG,.,.,//C,N,O/C,Y,N1=MMG $
LABEL LBL7 $
PARAM //C,N,NOP/V,Y,PCHMD=-1 $
COND  LBL8,PCHMD $
OUTPUT3 PHIG,.,.,//C,N,O/C,Y,N1=PHG $
LABEL LBL8 $
PARAM //C,N,NOP/V,Y,PCHMM=-1 $
COND  LBL9,PCHMM $
OUTPUT3 KHH,BHH,MHH,.,//C,N,O/C,Y,N1=KKH/C,Y,N2=BBH/C,Y,N3=MMH $
LABEL LBL9 $
JUMP  FINIS$
LABEL ERRO2 $
PRTPARM //C,N,-2/C,N,MODES $
LABEL FINIS $
END $
CEND

```

Because NASTRAN is extremely inflexible in permitting the user to control output data formatting, a preprocessor program must be available for reading the NASTRAN-generated data. The next program is designed to not only read the data but also to process the data and compute the input data required by both N-BOD2 and DISCOS in respectively acceptable format.

For DISCOS users, the mode-dependent parameters are outputted on the line printer and on magnetic tape (data set 2) in a DISCOS-compatible format.

```

C
C
C      DISCOS, N-BOD2 PREPROCESSOR OF NASTRAN GENERATED DATA
C
C      PURPOSE:
C          1) READ ONE OR MORE NASTRAN GENERATED TAPES
C          2) WRITE ONE INPUT TAPE FOR DISCOS PROGRAM
C             CONTAINING FLEXIBLE BODY DATA FOR ALL
C             FLEXIBLE BODIES IN SIMULATION
C          3) COMPUTE RESULTANT MODE DEPENDENT PARAMETERS
C             FOR GSFC MULTI-FLEXIBLE BODY PROGRAM
C             N-BOD2, THIS SECTION OF NO INTEREST TO
C             DISCOS USERS
C
C
C      DEFINITIONS:
C
C      DATA SET 1 = SCRATCH FILE USED TO PROCESS NASTRAN DATA
C
C      DATA SET 2 = DISCOS INPUT TAPE (THE OUTPUT TAPE OF THIS PROGRAM)
C      THE CARD OUTPUT OF THIS PROGRAM ASSUMES THAT DATA SET 14 IN
C      DISCOS WILL BE RESERVED FOR THE FLEXIBLE BODY DATA
C
C      DATA SET 3 = SCRATCH FILE USED TO REDUCE CORE REQUIREMENTS
C
C      DATA SET 11 = LOCATION IN SYSTEM OF NASTRAN TAPE ASSOCIATED WITH
C      THE FLEXIBLE BODY IN DISCOS HAVING LOWEST MAGNITUDE
C      INTEGER LABEL
C
C      DATA SET 12 = LOCATION IN SYSTEM OF NASTRAN TAPE ASSOCIATED WITH
C      THE NEXT FLEXIBLE BODY IN THE DISCOS MODEL
C
C      .
C      .
C      .
C      ETC.      PROGRAM DIMENSIONED FOR A MAXIMUM OF 6 NASTRAN TAPES
C
C      IMPLICIT REAL*8 (A-H,O-Z)
C
C      DATA IDL/1H$/
C      DATA ISTR,IBLK/1H*,1H /
C      DATA IGPL,IBGP,IMGG      /4HGPL ,4HBGPD,4HMGG /

```



```

DATA ID,IT,II/1HD,1HT,1HI/
DATA IR,IE,IS/1HR,1HE,1HS/
C
INTEGER INPT(80), IGRID(500), IA(20), IB(20),
* ICOT(5), IY(4)
INTEGER MSAVE(100),MSTOTL,HINGE
C
REAL*8 X(500), Y(500), Z(500),
* INERTA(6,6,500), PHI(6,500,12),
* KHH(12,12), BHH(12,12),
* MHH(12,12), XX(4), XY(4)
C
REAL*8 AMD(500,72)
C
COMMON /WORK/ AMD
C
EQUIVALENCE (AMD(1,1),PHI(1,1,1))
EQUIVALENCE (AMD(1,1),X(1))
EQUIVALENCE (AMD(1,2),Y(1))
EQUIVALENCE (AMD(1,3),Z(1))
EQUIVALENCE (AMD(1,4),INERTA(1,1,1))
C
LOGICAL LECHO,LPUNCH,LCHECK,LNAST
LOGICAL LRSTRT
C
C
100 FORMAT (I5,3L5,2I5,L5)
101 FORMAT (4A4,7I8,2A4)
102 FORMAT (2A4,8I8)
103 FORMAT (4A4,5I8,E16.9)
104 FORMAT (2A4,2E16.9,I16,E16.9)
105 FORMAT (2A4,2E16.9)
106 FORMAT (I10)
108 FORMAT (4A4,I8,2I16,E16.8)
109 FORMAT (10E13.5)
110 FORMAT (6E15.5)
111 FORMAT (12I5)
112 FORMAT ('I',I5,3L5,2I5,L5)
113 FORMAT (1X,20I5)
114 FORMAT ('I',5X,'***** MODAL DATA SELECTED FROM NASTRAN INPUT TAP
*E',I4,' FOR PROCESSING *****',/)
115 FORMAT (12X,'MODE',I4,' LISTED BELOW IS MODE',I4,' ON THE NASTRAN
*DATA TAPE')
200 FORMAT (' NJOINT =',I5,' NUMBER OF GRID POINT LOCATIONS TO BE READ
* =',I5,' THESE SHOULD BE EQUAL')
201 FORMAT (' NUMBER GRID POINT X-LOCATION Y-LOCATION
* Z-LOCATION ')
202 FORMAT (6X,I5,7X,I5,2X,3E15.5)
300 FORMAT (80A1)
302 FORMAT (4A4,I8,2I16,E16.8,2A4)
303 FORMAT (2A4,4E16.8,2A4)
400 FORMAT (2X,80A1)
500 FORMAT (' ANOMOLOUS EOF MARK ON TAPE, LISTING FOLLOWS ')
501 FORMAT (/, ' INTERNAL GRID POINT ',I5,' EXTERNAL GRID POINT',I5,' I
*N THE NASTRAN BULK DATA DECK')
502 FORMAT (' X,Y,Z COORDINATES')
503 FORMAT (6E20.10)
504 FORMAT (' LUMPED INERTIA MATRIX')
505 FORMAT (' PHI (6 MODAL COMPONENTS FOR EACH SELECTED MODE)')

```

```

506 FORMAT (//,' KHH ')
507 FORMAT ('1',5X,'***** LUMPED PARAMETER AND MODAL DATA TO BE PROCESSED FOR DISCOS AND N-ROD2 INPUT DATA *****',//)
508 FORMAT (//,' BHH ')
509 FORMAT (//,' MHH ')
510 FORMAT (////)
511 FORMAT (2X,'EXIT DECODE ',4(I8,E16.8))
512 FORMAT (2X,'ENTER DECODE ',2I8,4E16.8)
600 FORMAT (I5,3E20.8)

```

```

C
C
C      READ TWO INPUT CARDS PER NASTRAN TAPE
C      NTAPE = DATA SET WHERE TAPE IS TO
C              BE FOUND NTAPE=11,12,...
C      LECHO = .TRUE. PRINT ECHO OF PROCESSED DATA
C              = .FALSE. PRINT ONLY IF ERROR FOUND ON TAPE
C      LPUNCH = .TRUE. PUNCH PLOT DATA
C              = .FALSE. DON'T
C      LCHECK = .TRUE. PRINT DATA IN/OUT OF DECODE
C              = .FALSE. DON'T
C      HINGE = GRID POINT NUMBER OF HINGE POINT AS
C              DEFINED BY USER IN NASTRAN BULK DATA
C      MSTOTL = TOTAL NUMBER OF FLEXIBLE BODY MODES TO
C              PROCESS FOR DISCOS AND N-ROD2 DATA
C      LNAST = .TRUE. PRINT NASTRAN DATA INPUT
C              = .FALSE. DON'T
C      MSAVE = MODE SAVE ARRAY
C              MSAVE(K) = 0 SKIP MODE K DATA
C              K PROCESS MODE K DATA
C

```

```

      REWIND 2
1 CONTINUE
  READ(5,100,END=4) NTAPE,LECHO,LPUNCH,LCHECK,HINGE,MSTOTL,LNAST
  WRITE(6,112)      NTAPE,LECHO,LPUNCH,LCHECK,HINGE,MSTOTL,LNAST
  READ(5,111) (IA(I),I=1,MSTOTL)
  DO 11 I=1,100
11  MSAVE(I) = 0
     DO 12 I=1,MSTOTL
12  MSAVE(IA(I)) = 1
     WRITE(6,113) MSAVE
  REWIND NTAPE

```

```

C      LRSTRT = .FALSE. NO RESTART CARDS ON NTAPE
C              = .TRUE.  RSTART CARDS ON NTAPE

```

```

      ICNTRS = 0
      LRSTRT = .FALSE.
9 READ (NTAPE,300,END=8) INPT
  IF(LNAST) WRITE(6,400) INPT
C  CHECK FOR RESTART CARDS
  IF(IR.NE.INPT(1)) GO TO 9
  IF(IE.NE.INPT(2)) GO TO 9
  IF(IS.NE.INPT(3)) GO TO 9
  IF(IT.NE.INPT(4)) GO TO 9
C  COME HERE FOR RESTART CARDS, COUNT THEM AND SET FLAG
  LRSTRT = .TRUE.
  ICNTRS = ICNTRS+1
13 READ (NTAPE,300,END=8) INPT
  IF(LNAST) WRITE(6,400) INPT
  IF(ID.EQ.INPT(1).AND.IT.EQ.INPT(2).AND.II.EQ.INPT(3)) GO TO 9
  ICNTRS = ICNTRS+1
  GO TO 13

```

```

      8 CONTINUE
      REWIND NTAPE
      REWIND 1
      REWIND 3
      ICNT = 0
      IBLK = 0
      DO 7 I=1,5
      7 ICOT(I) = 0

C
C      READ ALL DTI CARDS FROM NASTRAN TAPE NTAPE TO OBTAIN
C      1) GRID POINT NUMBERING SEQUENCE SET UP BY NASTRAN
C      2) GRID POINT LOCATIONS
C
C      **** NOTE ****
C
C      NASTRAN WILL RENUMBER GRID POINTS DEFINED BY USER
C      INTO A SEQUENTIAL SET, THIS IS USED FOR INTERNAL NASTRAN
C      COMPUTATION AND WILL BE USED BY DISCOS
C
C      SKIP RESTART CARDS IN FRONT OF GOOD DATA
C      IF(.NOT.LRSTRT) GO TO 2
C      DO 14 I=1,ICNTRS
14  READ (NTAPE,300,END=3) INPT
C      2 READ (NTAPE,101,END=3) (IA(I),I=1,4),(IB(I),I=1,7),(IA(I),I=5,6)
C      CHECK FOR TABLE HEADING
C      IF(IA(3).EQ.IGPL) GO TO 50
C      IF(IA(3).EQ.IBGP) GO TO 60
C      IF(IA(3).EQ.IMGG) GO TO 70
C      GO TO 2
C
C      ERROR ON TAPE
C      3 CONTINUE
C      WRITE(6,500)
C      6 REWIND NTAPE
C      5 READ (NTAPE,300,END=4) INPT
C      WRITE(6,400) INPT
C      GO TO 5
C      4 STOP
C
C      COME HERE TO PROCESS GPL TABLE DATA
C      50 CONTINUE
C      ICNT = ICNT + 1
C      IF(ICNT.NE.1) GO TO 51
C      NJOINT = IB(2)
C
C      NJOINT = NUMBER OF GRID POINTS IN MODEL
C      GO TO 2
C      51 IF(ICNT.NE.2)GO TO 52
C      DO 53 J=1,6
C      53 IGRID(J) = IB(J+1)
C      NC = (NJOINT-6)/8
C      DO 54 J=1,NC
C      JJ = 7 + 8*(J-1)
C      JJ7 = JJ+7
C      54 READ(NTAPE,102) (IA(I),I=1,2),(IGRID(I),I=JJ,JJ7)
C      JJ = 6 + 8*NC
C      JJ1 = JJ+1
C      IF(JJ.EQ.NJOINT) GO TO 2
C      READ(NTAPE,102) (IA(I),I=1,2),(IGRID(I),I=JJ1,NJOINT)
C      GO TO 2
C
C      IGRID(I) =USER DEFINED GRID POINT NUMBER

```

```

C                                     WHICH NASTRAN HAS RELABLED TO
C                                     RE INTERNAL NUMBER I
52 ICNT = 0
   NC = (NJOINT-3)/4
   IF(NJOINT.EQ.4*NC+3) GO TO 16
   NC = NC + 1
16 DO 15 I=1,NC
15 READ (NTAPE,300) INPT
   GO TO 2

C
C
C                                     COME HERE TO PROCESS HGPDT CARDS
60 CONTINUE
   IF(1B(2).EQ.NJOINT) GO TO 61
C   ERROR
   WRITE(6,200) NJOINT,1B(2)
   GO TO 6
61 READ (NTAPE,300) INPT
   READ (NTAPE,103) (1A(I),I=1,4),(1B(I),I=1,5),X(1)
   DO 62 J=2,NJOINT
62 READ (NTAPE,104) (1A(I),I=1,2),Y(J-1),Z(J-1),1B(1),X(J)
   J = NJOINT
   READ (NTAPE,105) (1A(I),I=1,2),Y(J),Z(J)
   GO TO 2
C   SAVE LOCATION DATA, WRITE IT ON TAPE 3 AFTER INERTIA DATA
C
C       X(J) = X-COORDINATE OF GRID POINT IGRID(J)
C       Y(J) = Y-COORDINATE OF GRID POINT IGRID(J)
C       Z(J) = Z-COORDINATE OF GRID POINT IGRID(J)
C
C
C                                     COME HERE TO PROCESS ALL DMI CARDS
C   STEP 1:
C       COUNT CARDS IN EACH GROUP
C       ASCERTAIN FORMAT OF EACH CARD
C       REWRITE ON SCRATCH FILE ALL CARDS
C       PRECEDED BY THEIR FORMAT CODE
C   STEP 2:
C       REWIND SCRATCH FILE
C       READ CARDS OFF SCRATCH FILE WITH KNOWN FORMAT
C       LOAD ALL ARRAYS, SKIPPING ALL UNWANTED MODES
70 CONTINUE
   REWIND 1
   IFORM = 301
   WRITE(1,106) IFORM
   WRITE(1,101) (1A(I),I=1,4),(1B(I),I=1,7),(1A(I),I=5,6)
   1BLK = 1
   ICOT(1BLK) = 1
   IF(1B(2).NE.6.OR.1B(6).NE.1B(7).OR.1B(6).NE.NJOINT*6) GO TO 71
   DO 76 I=1,NJOINT
   DO 76 J1=1,6
   DO 76 J2=1,6
76 INERTA(J1,J2,I) =0.0
   GO TO 72
71 CONTINUE
C   ERROR
   WRITE(6,101) (1A(I),I=1,4),(1B(I),I=1,7),(1A(I),I=5,6)
   GO TO 6
72 READ(NTAPE,300,END=80) INPT
   IF(INPT(4).NE.1STR) GO TO 73

```

```

        ICOT(IBLK) = ICOT(IBLK) + 1
        IFORM = 302
        GO TO 74
73 IF(INPT(1).EQ.ISTR) GO TO 75
C      END OF DATA BLOCK IBLK
C      CHECK FOR 'END OF CHECKPOINT DICTIONARY'
        IF(INPT(1).EQ.IDL) GO TO 72
        IBLK = IBLK + 1
        ICOT(IBLK) = ICOT(IBLK) + 1
        IFORM = 301
        GO TO 74
75 CALL INTERP(INPT,IFORM)
        ICOT(IBLK) = ICOT(IBLK) + 1
74 WRITE(1,106) IFORM
        WRITE(1,300) INPT
        GO TO 72
80 END FILE 1

C
C      ALL DMI CARDS READ, FORMAT DEDUCED AND
C      REWRITTEN ON SCRATCH FILE 1
C      FORMAT CODES:
C      301 = FORMAT (4A4,7I8,2A4)
C      302 = FORMAT (4A4,I8,2I16,E16.8,2A4)
C      303 = FORMAT (2A4,4E16.8,2A4)
C      304 = FORMAT (2A4,3E16.8,I16,2A4)
C      305 = FORMAT (2A4,2E16.8,I16,E16.8,2A4)
C      306 = FORMAT (2A4,E16.8,I16,2E16.8,2A4)
C      307 = FORMAT (2A4,E16.8,I16,E16.8,I16,2A4)
C      308 = FORMAT (2A4,I16.8,3E16.8,2A4)
C      309 = FORMAT (2A4,I16.8,2E16.8,I16.8,2A4)
C      310 = FORMAT (2A4,I16,E16.8,I16,E16.8,2A4)
C
C      START TO PROCESS DMI CARDS
C      REWIND 1
        IF(IBLK.LE.5) GO TO 20
C      ERROR IBLK MUST BE LESS THAN OR EQUAL TO 5
81 REWIND 1
83 READ (1,300,END=90) INPT
        WRITE(6,400) INPT
        GO TO 83
90 STOP
20 CONTINUE
        DO 22 IBK=1,IBLK
            ICBLK = ICOT(IBK)
            DO 21 JJ =1,ICBLK
                IF(JJ.NE.1) GO TO 23
C      READ FIRST CARD OF DATA BLOCK
                READ (1,106) IFORM
                READ (1,101) (IA(I),I=1,4),(IB(I),I=1,7),(IC(I),I=5,6)
                IF(IBK.NE.1) GO TO 40
C
C      INERTIA MATRIX
                IF(IB(2).NE.6.OR.IB(6).NE.IB(7).OR.IB(6).NE.NJOINT*6) GO TO 81
C      SO FAR SO GOOD
                GO TO 21
40 IF(IBK.NE.2) GO TO 41
C
C      MODE SHAPES
                IF(IB(6).NE.6*NJOINT) GO TO 81

```

```

      NMODES = IB(7)
C
C      NMODES = NUMBER OF MODES
      MSTOTL = NUMBER OF MODES TO PROCESS
      DO 42 I1=1,6
      DO 42 I2=1,NJOINT
      DO 42 I3=1,MSTOTL
42  PHI(I1,I2,I3) = 0.0
      GO TO 21
C
C      MODAL MASS, STIFFNESS AND DAMPING MATRICES
41  IF(IB(6).NE.IB(7).OR.IB(7).NE.NMODES) GO TO 71
      DO 43 I1=1,MSTOTL
      DO 43 I2=1,MSTOTL
      IF(IBK.NE.3) GO TO 44
      KHH(I1,I2) = 0.0
      GO TO 43
44  IF(IBK.NE.4) GO TO 45
      BHH(I1,I2) = 0.0
      GO TO 43
45  MHH(I1,I2) = 0.0
43  CONTINUE
      GO TO 21
C
C      READ IN DATA CARDS
23  CONTINUE
      READ (1,106) IFORM
      IF(IFORM.NE.302) GO TO 24
      READ (1,302) (IA(I),I=1,4),IB(1),NNR,NNC,XX(1),(IA(I),I=5,6)
      NSR = MSAVE(NNR)
      NSC = MSAVE(NNC)
      IF(IBK.NE.1) GO TO 25
      N1T = (NNC-1)/6 + 1
      N2T = (NNR-1)/6 + 1
      IF(N1T.EQ.N2T) GO TO 26
C      ERROR
      WRITE(6,302) (IA(I),I=1,4),IB(1),NNR,NNC,XX(1),(IA(I),I=5,6)
      GO TO 81
26  NT = N1T
      NC = MOD(NNC-1,6) + 1
      NR = MOD(NNR-1,6) + 1
      INERTA(NR,NC,NT) = XX(1)
      GO TO 21
25  IF(IBK.NE.2) GO TO 27
      NT = (NNC-1)/6 + 1
      NE = MOD(NNC-1,6) + 1
      IF(NSR.EQ.0) GO TO 21
      PHI(NE,NT,NSR) = XX(1)
      GO TO 21
27  IF(IBK.NE.3) GO TO 28
      IF(NSR.EQ.0.OR.NSC.EQ.0) GO TO 21
      KHH(NSR,NSC) = XX(1)
      GO TO 21
28  IF(IBK.NE.4) GO TO 29
      IF(NSR.EQ.0.OR.NSC.EQ.0) GO TO 21
      BHH(NSR,NSC) = XX(1)
      GO TO 21
29  CONTINUE
      IF(NSR.EQ.0.OR.NSC.EQ.0) GO TO 21
      MHH(NSR,NSC) = XX(1)
      GO TO 21

```

```

C
24 CONTINUE
C      FORMAT 4E16.8 FOR XX WILL READ INTEGER I WHICH IS RIGHT
C      JUSTIFIED AS I * 1.0E-08
C      THIS IS EASILY UNDONE IN DECODE
READ (1,303) (IA(I),I=1,2),(XX(I),I=1,4),(IA(I),I=3,4)
IF(LCHECK) WRITE (6,512) IFORM,NNC,(XX(I),I=1,4)
CALL DECODE(IFORM,XX,NNC,XY,IY,NY)
IF(LCHECK) WRITE (6,511) (IY(I),XY(I),I=1,NY)
DO 30 I=1,NY
  IF(IBK.NE.1) GO TO 31
  NC = MOD(IY(I)-1,6) + 1
  INERTA(NR,NC,NT) = XY(I)
  GO TO 30
31 IF(IBK.NE.2) GO TO 32
  NT = (IY(I)-1)/6 + 1
  NE = MOD(IY(I)-1,6) + 1
  IF(NSR.EQ.0) GO TO 30
  PHI(NE,NT,NSR) = XY(I)
  GO TO 30
32 IF(IBK.NE.3) GO TO 33
  NSC = MSAVE(IY(I))
  IF(NSR.EQ.0.OR.NSC.EQ.0) GO TO 30
  KHH(NSR,NSC) = XY(I)
  GO TO 30
33 IF(IBK.NE.4) GO TO 34
  NSC = MSAVE(IY(I))
  IF(NSR.EQ.0.OR.NSC.EQ.0) GO TO 30
  BHH(NSR,NSC) = XY(I)
  GO TO 30
34 CONTINUE
  NSC = MSAVE(IY(I))
  IF(NSR.EQ.0.OR.NSC.EQ.0) GO TO 30
  MHH(NSR,NSC) = XY(I)
30 CONTINUE
21 CONTINUE
C
C      IF(IBK.NE.1) GO TO 63
C      REWIND 3
C      WRITE (3) (((INERTA(I,J,N),I=1,6),J=1,6),N=1,NJOINT)
C      WRITE (3) (X(N),Y(N),Z(N),N=1,NJOINT)
C      IF(.NOT.LPUNCH) GO TO 10
C      WRITE (7,600) (IGRID(N),X(N),Y(N),Z(N),N=1,NJOINT)
10 CONTINUE
  WRITE(6,507)
  DO 64 N=1,NJOINT
    WRITE (6,501) N,IGRID(N)
    WRITE (6,502)
    WRITE (6,503) X(N),Y(N),Z(N)
    WRITE (6,504)
    WRITE (6,503) ((INERTA(I,J,N),J=1,6),I=1,6)
64 CONTINUE
  WRITE (6,510)
  GO TO 22
63 IF(IBK.NE.2) GO TO 65
  WRITE (3) (((PHI(I,N,J),I=1,6),J=1,MSTOTL),N=1,NJOINT)
  IF(.NOT.LPUNCH) GO TO 67
  DO 68 J=1,MSTOTL
    WRITE (7,600) (IGRID(N),(PHI(I,N,J),I=1,3),N=1,NJOINT)

```

```

68 CONTINUE
67 CONTINUE
  WRITE (6,114) NTAPE
  DO 17 I=1,NMODES
    IF(MSAVE(I).EQ.0) GO TO 17
    WRITE (6,115) MSAVE(I),I
17 CONTINUE
  WRITE (6,510)
  DO 66 N=1,NJOINT
    WRITE (6,501) N,IGRID(N)
    WRITE (6,505)
    WRITE (6,503) ((PHI(I,N,J),I=1,6),J=1,MSTOTL)
66 CONTINUE
  WRITE (6,510)
  GO TO 22
65 IF(IBK.NE.3) GO TO 69
  WRITE (3) ((KHH(I,J),I=1,MSTOTL),J=1,MSTOTL)
  WRITE (6,506)
  WRITE (6,503) ((KHH(I,J),J=1,MSTOTL),I=1,MSTOTL)
  WRITE (6,510)
  GO TO 22
69 IF(IBK.NE.4) GO TO 77
  WRITE (3) ((BHH(I,J),I=1,MSTOTL),J=1,MSTOTL)
  WRITE (6,508)
  WRITE (6,503) ((BHH(I,J),J=1,MSTOTL),I=1,MSTOTL)
  WRITE (6,510)
  GO TO 22
77 IF(IBK.NE.5) GO TO 22
  WRITE (3) ((MHH(I,J),I=1,MSTOTL),J=1,MSTOTL)
  WRITE (6,509)
  WRITE (6,503) ((MHH(I,J),J=1,MSTOTL),I=1,MSTOTL)
  WRITE (6,510)
22 CONTINUE
C
C          DATA READY FOR DISCOS PROCESSING
  CALL DISCOS(IGRID,NJOINT,MSTOTL,ITAPE,LECHO)
  CALL NBOD2 (IGRID,NJOINT,MSTOTL,ITAPE,LECHO,HINGE)
  IF(ITAPE.LE.6) GO TO 1
  STOP
  END

```

```

SUBROUTINE DISCOS(IGRID,NJ,NM,ICNT,LECHO)
  IMPLICIT REAL*8 (A-H,O-Z)

```

```

C
  LOGICAL LECHO
C

```

```

  REAL*8    X(500),Y(500),Z(500),INERTA(6,6,500),IRUNNO
  REAL*8    PHI(6,500,12),KHH(12,12),BHH(12,12),LOC(6)
  REAL*8    AMS(6),JMASS(500,1),JINER(500,6),AIN(6),SMM(6)
  REAL*8    SMSM(500,3),LOCA(500,3),MOD(6,6),AMD(500,13),KH(6),BH(6)
  REAL*8    AMP(500,72)
  COMMON /WORK/ AMP
  EQUIVALENCE (AMP(1,1),X(1))
  EQUIVALENCE (AMP(1,2),Y(1))
  EQUIVALENCE (AMP(1,3),Z(1))

```



```

EQUIVALENCE (AMP(1,4),INERTA(1,1,1))
EQUIVALENCE (AMP(1,1),PHI(1,1,1))
C
INTEGER IGRID(500)
C
COMMON /WORK2/ AMD
EQUIVALENCE (AMD(1,1),JMASS(1,1)), (AMD(1,1),JINER(1,1))
EQUIVALENCE (AMD(1,7),SMSM(1,1)), (AMD(1,10),LOCA(1,1))
EQUIVALENCE (AMD(1,13),KHH(1,1)), (AMD(145,13),BHH(1,1))
C
DATA IFST/0/
DATA AMS/6HJMASS1,6HJMASS2,6HJMASS3,6HJMASS4,6HJMASS5,
* 6HJMASS6/
DATA FLX/6HFLXDAT/
DATA AIN/6HINERT1,6HINERT2,6HINERT3,6HINERT4,6HINERT5,
* 6HINERT6/
DATA SMM/6HSTMSM1,6HSTMSM2,6HSTMSM3,6HSTMSM4,6HSTMSM5,
* 6HSTMSM6/
DATA LOC/6HLOCAT1,6HLOCAT2,6HLOCAT3,6HLOCAT4,6HLOCAT5,
* 6HLOCAT6/
DATA MOD/6HXDIS1,6HYDIS1,6HZDIS1,6HTHA11,6HTHA21,6HTHA31,
* 6HXDIS2,6HYDIS2,6HZDIS2,6HTHA12,6HTHA22,6HTHA32,
* 6HXDIS3,6HYDIS3,6HZDIS3,6HTHA13,6HTHA23,6HTHA33,
* 6HXDIS4,6HYDIS4,6HZDIS4,6HTHA14,6HTHA24,6HTHA34,
* 6HXDIS5,6HYDIS5,6HZDIS5,6HTHA15,6HTHA25,6HTHA35,
* 6HXDIS6,6HYDIS6,6HZDIS6,6HTHA16,6HTHA26,6HTHA36 /
DATA KH/6HKHH1,6HKHH2,6HKHH3,6HKHH4,6HKHH5,6HKHH6 /
DATA BH/6HBHH1,6HBHH2,6HBHH3,6HBHH4,6HBHH5,6HBHH6 /
DATA IRUNNO/6HPREPRS/
C
700 FORMAT (A6,' 0 14PREPRS ')
C
C INITIALIZE TAPE, START COUNTER
REWIND 3
IF(IFST.NE.0) GO TO 1
CALL INTAPE(2,FLX)
ICNT = 0
IFST = 1
1 CONTINUE
ICNT = ICNT+1
C
C PROCESS MASS MATRIX DATA
READ (3) (((INERTA(I,J,N),I=1,6),J=1,6),N=1,NJ)
READ (3) (X(N),Y(N),Z(N),N=1,NJ)
DO 2 I=1,NJ
IF(INERTA(1,1,I).NE.INERTA(2,2,I)) GO TO 3
IF(INERTA(2,2,I).NE.INERTA(3,3,I)) GO TO 3
IF(INERTA(3,3,I).LT.0) GO TO 3
JMASS(I,1) = INERTA(3,3,I)
GO TO 2
3 WRITE(6,500) I,(INERTA(II,II,I),II=1,3)
500 FORMAT (' ERROR MASS MATRIX POINT ',I5,' JMASS = ',3E20.10)
JMASS(I,1) = (INERTA(1,1,I)+INERTA(2,2,I)+INERTA(3,3,I))/3.000
2 CONTINUE
C
C PUNCH READ MATRIX INPUT CARD
C PUT MASS MATRIX ON TAPE
WRITE(7,700) AMS(ICNT)
IF(LECHO) CALL WRITE(JMASS,NJ,1,AMS(ICNT),500)
CALL WTAPE(JMASS,NJ,1,AMS(ICNT),500,2)
C

```

```

C          PROCESS INERTIA DYAD DATA
DO 4 I=1,NJ
  JINER(I,1) = INERTA(4,4,I)
  JINER(I,2) = INERTA(5,5,I)
  JINER(I,3) = INERTA(6,6,I)
  JINER(I,4) = -INERTA(4,5,I)
  JINER(I,5) = -INERTA(4,6,I)
4 JINER(I,6) = -INERTA(5,6,I)
  WRITE(7,700) AIN(ICNT)
  IF(LECHO) CALL WRITE(JINER,NJ,6,AIN(ICNT),500)
  CALL WTAPE(JINER,NJ,6,AIN(ICNT),500,2)

C          PROCESS STATIC MASS MOMENT DATA
C          PROCESS JOINT LOCATION DATA
DO 5 I=1,NJ
  SMSM(I,1) = INERTA(2,6,I)
  SMSM(I,2) = -INERTA(1,6,I)
  SMSM(I,3) = INERTA(1,5,I)
  LOCA(I,1) = X(I)
  LOCA(I,2) = Y(I)
5 LOCA(I,3) = Z(I)
  WRITE(7,700) SMM(ICNT)
  IF(LECHO) CALL WRITE(SMSM ,NJ,3,SMM(ICNT),500)
  CALL WTAPE(SMSM,NJ,3,SMM(ICNT),500,2)
  WRITE(7,700) LOC(ICNT)
  IF(LECHO) CALL WRITE(LOCA ,NJ,3,LOC(ICNT),500)
  CALL WTAPE(LOCA,NJ,3,LOC(ICNT),500,2)

C          PROCESS MODAL DATA
C          READ (3) ((PHI(I,N,J),I=1,6),J=1,NM),N=1,NJ)
DO 6 K=1,6
DO 7 I=1,NJ
DO 7 M=1,NM
7 AMD(I,M) = PHI(K,I,M)
  WRITE(7,700) MOD(K,ICNT)
  IF(LECHO) CALL WRITE(AMD,NJ,NM,MOD(K,ICNT),500)
  CALL WTAPE(AMD,NJ,NM,MOD(K,ICNT),500,2)
6 CONTINUE

C          MODAL MATRICES
C          READ (3) ((KHH(I,J),I=1,NM),J=1,NM)
C          READ (3) ((BHH(I,J),I=1,NM),J=1,NM)
WRITE(7,700) KH(ICNT)
IF(LECHO) CALL WRITE(KHH,NM,NM,KH(ICNT),12)
CALL WTAPE(KHH,NM,NM,KH(ICNT),12,2)
WRITE(7,700) BH(ICNT)
WRITE(6,100)
IF(LECHO) CALL WRITE(BHH,NM,NM,BH(ICNT),12)
CALL WTAPE(BHH,NM,NM,BH(ICNT),12,2)

C          CALL LTAPE(2)
CALL RTAPE(IRUNNO,AMS(ICNT),AMD,NJ,1,500,12,2)
CALL WRITE(AMD,NJ,1,AMS(ICNT),500)
CALL RTAPE(IRUNNO,AIN(ICNT),AMD,NJ,6,500,12,2)
CALL WRITE(AMD,NJ,6,AIN(ICNT),500)
CALL RTAPE(IRUNNO,SMM(ICNT),AMD,NJ,3,500,12,2)
CALL WRITE(AMD,NJ,3,SMM(ICNT),500)
CALL RTAPE(IRUNNO,LOC(ICNT),AMD,NJ,3,500,12,2)
CALL WRITE(AMD,NJ,3,LOC(ICNT),500)
DO 8 K=1,6

```

```

      CALL RTAPE(IRUNNO,MOD(K,ICNT),AMD,NJ,NM,500,12,2)
8  CALL WRITE(AMD,NJ,NM,MOD(K,ICNT),500)
      CALL RTAPE(IRUNNO,KH(ICNT),KHH,NM,NM,12,12,2)
      CALL WRITE(KHH,NM,NM,KH(ICNT),12)
      CALL RTAPE(IRUNNO,BH(ICNT),BHH,NM,NM,12,12,2)
      CALL WRITE(BHH,NM,NM,BH(ICNT),12)
100 FORMAT ('1',5X,'***** PROCESSED LUMPED PARAMETER AND MODAL DATA N
*OW ON DATA SET 2 (DISCOS INPUT TAPE) *****',///)
      RETURN
      END

```

```

C      SUBROUTINE NBOD2 (IGRID,NJ,NM,ICNT,LECHO,HINGE)
C      COMPUTE RESULTANT MODE DEPENDENT PARAMETERS FOR N-BOD2
C      IMPLICIT REAL*8 (A-H,O-Z)
C
C      LOGICAL LECHO
C
C      COMMON /WORK/  AMP
C      REAL*8 AMP(500,72)
C      COMMON /WORK2/ AMD
C      REAL*8 AMD(500,13)
C
C      REAL*8  AMS(6),FLX,AIN(6),SMM(6),LOC(6),MOD(6,6),KH(6),BH(6)
C      REAL*8  JMASS(500),JLOC(3,500),CMLOC(3),JINERT(3,3,500),
C      *      PHIT(3,500,12),PHIR(3,500,12),SMO(3,500),
C      *      FLOM(12),ZETA(12),BDINER(3,3),
C      *      FLA(3,12),FLB(3,12),FLC(3,12),FLD(3,3,12),
C      *      FLJ(3,3,12),
C      *      FCF(3,3,12,12),FCK(3,12,12),
C      *      TEM(3),TEM1(3,3),ADM(500,12,6),TFM(3),
C      *      KHH(12,12),BHH(12,12),IRUNNO
C      INTEGER IGRID(500)
C      INTEGER HINGE,INTH
C
C      DATA  AMS/6HJMASS1,6HJMASS2,6HJMASS3,6HJMASS4,6HJMASS5,
C      *      6HJMASS6/
C      DATA  AIN/6HINERT1,6HINERT2,6HINERT3,6HINERT4,6HINERT5,
C      *      6HINERT6/
C      DATA  SMM/6HSTMSM1,6HSTMSM2,6HSTMSM3,6HSTMSM4,6HSTMSM5,
C      *      6HSTMSM6/
C      DATA  LOC/6HLOCAT1,6HLOCAT2,6HLOCAT3,6HLOCAT4,6HLOCAT5,
C      *      6HLOCAT6/
C      DATA  MOD/6HXDIS1 ,6HYDIS1 ,6HZDIS1 ,6HTHA11 ,6HTHA21 ,6HTHA31 ,
C      *      6HXDIS2 ,6HYDIS2 ,6HZDIS2 ,6HTHA12 ,6HTHA22 ,6HTHA32 ,
C      *      6HXDIS3 ,6HYDIS3 ,6HZDIS3 ,6HTHA13 ,6HTHA23 ,6HTHA33 ,
C      *      6HXDIS4 ,6HYDIS4 ,6HZDIS4 ,6HTHA14 ,6HTHA24 ,6HTHA34 ,
C      *      6HXDIS5 ,6HYDIS5 ,6HZDIS5 ,6HTHA15 ,6HTHA25 ,6HTHA35 ,
C      *      6HXDIS6 ,6HYDIS6 ,6HZDIS6 ,6HTHA16 ,6HTHA26 ,6HTHA36 /
C      DATA  KH/6HKHH1 ,6HKHH2 ,6HKHH3 ,6HKHH4 ,6HKHH5 ,6HKHH6 /
C      DATA  BH/6HBHH1 ,6HBHH2 ,6HBHH3 ,6HBHH4 ,6HBHH5 ,6HBHH6 /
C      DATA  IRUNNO/6HPREPRS/
C
C      EQUIVALENCE (AMP(1,1),PHIT(1,1,1)),(AMP(1,37),PHIR(1,1,1)),
C      *      (AMP(1,1),ADM(1,1,1)),
C      *      (AMP(1,13),ADM(1,1,2)),

```

```

*          (AMP(1,25),ADM(1,1,3)),
*          (AMP(1,37),ADM(1,1,4)),
*          (AMP(1,49),ADM(1,1,5)),
*          (AMP(1,61),ADM(1,1,6))
EQUIVALENCE (AMD(1,1),JMASS(1)), (AMD(1,2),JLOC(1,1)),
*          (AMD(1,5),SMO(1,1)), (AMD(1,5),JINERT(1,1,1))

C
C      FIND INTERNAL NASTRAN NUMBER ASSOCIATED
C      WITH THE HINGE POINT (GRID POINT 'HINGE')
      DO 1 N=1,NJ
      IF(IGRID(N).EQ.HINGE) GO TO 2
1  CONTINUE
      WRITE(6,100) HINGE
      RETURN
2  INTH = N

C
C          INTH = INTERNAL NASTRAN NUMBER
C          OF HINGE POINT

C
C      GET DATA OFF OF TAPE 2 TO COMPUTE CENTER OF MASS

      CALL RTAPE(IRUNNO,LOC(ICNT),AMP,NJ,3,500,72,2)

C
C          JLOC(I,N) = I-TH COORDINATE OF GRID POINT IGRID(N)
C          VECTOR RELATIVE TO HINGE POINT HINGE

      DO 3 N=1,NJ
      DO 3 I=1,3
      JLOC(I,N) = AMP(N,I) - AMP(INTH,I)
3  CONTINUE

C
      CALL RTAPE(IRUNNO,AMS(ICNT),AMP,NJ,1,500,72,2)

C
C          JMASS(N) = LUMPED MASS AT GRID POINT IGRID(N)

      DO 4 N=1,NJ
      JMASS(N) = AMP(N,1)
4  CONTINUE

C
      IF(.NOT.LECHO) GO TO 17
      WRITE(6,101)
      WRITE(6,102) (N,IGRID(N),N,(JLOC(I,N),I=1,3),N,JMASS(N),N=1,NJ)
17 CONTINUE

C
C      COMPUTE CENTER OF MASS LOCATION

      TOTMAS = 0.0
      CMLOC(1) = 0.0
      CMLOC(2) = 0.0
      CMLOC(3) = 0.0

C
C      READ GRID POINT INERTIA TENSORS AND MASS MOMENTS

      CALL RTAPE(IRUNNO,SMM(ICNT),AMP,NJ,3,500,72,2)
      DO 11 N=1,NJ
      DO 11 I=1,3
      SMO(I,N) = AMP(N,I)
      IF(SMO(I,N).EQ.0.0) GO TO 11
      WRITE(6,107) (N,(SMO(I,N),I=1,3))

```

```

11 CONTINUE
   IF(LECHO) CALL WRITE(SMO,3,NJ,6H SMO  ,3)
C
   CALL RTAPE(IRUNNO,AIN(ICNT),AMP,NJ,6,500,72,2)
   DO 10 N=1,NJ
     JINERT(1,1,N) = AMP(N,1)
     JINERT(1,2,N) = -AMP(N,4)
     JINERT(1,3,N) = -AMP(N,5)
     JINERT(2,1,N) = -AMP(N,4)
     JINERT(2,2,N) = AMP(N,2)
     JINERT(2,3,N) = -AMP(N,6)
     JINERT(3,1,N) = -AMP(N,5)
     JINERT(3,2,N) = -AMP(N,6)
     JINERT(3,3,N) = AMP(N,3)
10 CONTINUE
   IF(LECHO) CALL WRITE(JINERT,9,NJ,6HJINERT,9)
C
C
C
C   COMPUTE UNDEFORMED BODY INERTIA TENSOR
C   RELATIVE TO BODY CENTER OF MASS
C
   DO 20 I=1,3
     DO 20 J=1,3
20  BDINER(I,J) = 0.0
     DO 19 N=1,NJ
       CALL VEC SUB(JLOC(1,N),CMLOC,TEM)
       CALL SUEOP(TEM,TEM,JMASS(N),TEM1)
       CALL DYADD(BDINER,TEM1,BDINER)
       CALL DYADD(BDINER,JINERT(1,1,N),BDINER)
19 CONTINUE
C
C
C   READ IN MODE SHAPES AND RESORT THEM
C
   DO 6 I=1,6
     CALL RTAPE(IRUNNO,MOD(I,ICNT),ADM(1,1,I),NJ,NM,500,12,2)
6  CONTINUE
     REWIND 3
     WRITE (3) (((ADM(N,M,I),I=1,3),N=1,NJ),M=1,NM)
     WRITE (3) (((ADM(N,M,I),I=4,6),N=1,NJ),M=1,NM)
     REWIND 3
     READ (3) (((PHIT(I,N,M),I=1,3),N=1,NJ),M=1,NM)
     READ (3) (((PHIR(I,N,M),I=1,3),N=1,NJ),M=1,NM)
     REWIND 3
C
     IF(.NOT.LECHO) GO TO 18
     WRITE(6,104)
     DO 7 M=1,NM
       WRITE(6,105)
       WRITE(6,106)      (N,M,(PHIT(I,N,M),I=1,3),
*                        N,M,(PHIR(I,N,M),I=1,3),N=1,NJ)
7  CONTINUE
18 CONTINUE
C
   WRITE(6,118)

```

```

WRITE(6,103) TOTMAS
WRITE(6,115)
WRITE(6,119) (BDINER(1,J),J=1,3)
WRITE(6,120) (BDINER(2,J),J=1,3)
WRITE(6,121) (BDINER(3,J),J=1,3)
WRITE(6,115)
WRITE(6,122) (CMLOC(I),I=1,3)
WRITE(6,115)
WRITE (6,125) INTH, HINGE
WRITE (6,115)

C
C      COMPUTE RESULTANT PARAMETERS
C
      DO 8 M=1,NM
      DO 8 I=1,3
      FLA(I,M) = 0.0
      FLB(I,M) = 0.0
      FLC(I,M) = 0.0
      DO 8 J=1,3
      FLJ(I,J,M) = 0.0
      FLD(I,J,M) = 0.0
8 CONTINUE
      DO 9 M=1,NM
      DO 9 MM=1,NM
      DO 9 I=1,3
      FCK(I,M,MM) = 0.0
      DO 9 J=1,3
      FCF(I,J,M,MM) = 0.0
9 CONTINUE

C
C      AMASS = 1.0/TOTMAS
C
      DO 12 M=1,NM
      DO 13 J=1,NJ

C      CALL SCLV(JMASS(J),PHIT(1,J,M),TEM)
      CALL VECADD(FLA(1,M),TEM,FLA(1,M))

C      CALL VEC SUB(JLOC(1,J),CMLOC,TEM)
      CALL SUEOP(PHIT(1,J,M),TEM,JMASS(J),TEM1)
      CALL DYADD(FLD(1,1,M),TEM1,FLD(1,1,M))

C      CALL VECROS(JLOC(1,J),PHIT(1,J,M),TEM)
      CALL SCLV(JMASS(J),TEM,TEM)
      CALL VECADD(FLB(1,M),TEM,FLB(1,M))

C      IF(JINERT(1,1,J).EQ.0.0.AND.
*      JINEPT(2,2,J).EQ.0.0.AND.
*      JINERT(3,3,J).EQ.0.0) GO TO 13

C      CALL DYDOTV(JINERT(1,1,J),PHIR(1,J,M),TEM)
      CALL VECADD(FLC(1,M),TEM,FLC(1,M))

C      CALL VECXDY(PHIR(1,J,M),JINERT(1,1,J),TEM1)
      CALL DYADD(FLJ(1,1,M),TEM1,FLJ(1,1,M))
13 CONTINUE

C
C      RESULTANT UNCOUPLED MODE DEPENDED PARAMETERS
C

```

```

      CALL SCLV(AMASS,FLA(1,M),FLA(1,M))
      CALL SCLV(AMASS,FLB(1,M),FLB(1,M))
12  CONTINUE
C
C
C
C
C      READ MODAL STIFFNESS AND DAMPING DATA
C
      CALL RTAPE(IRUNNO,KH(ICNT),KHH,NM,NM,12,12,2)
      CALL RTAPE(IRUNNO,BH(ICNT),BHH,NM,NM,12,12,2)
C
      DO 16 M=1,NM
      WRITE (6,116) M
      FLOM(M) = DSQRT(KHH(M,M))
      ZETA(M) = BHH(M,M)/(2.0*FLOM(M))
      WRITE(6,115)
      WRITE(6,123) M,FLOM(M)
      WRITE(6,115)
      WRITE(6,124) M,ZETA(M)
      WRITE(6,115)
      WRITE(6,108) (M,(FLA(I,M),I=1,3))
      WRITE(6,115)
      WRITE(6,109) (M,(FLB(I,M),I=1,3))
      WRITE(6,115)
      WRITE(6,110) (M,(FLC(I,M),I=1,3))
      WRITE(6,115)
      WRITE(6,126) (M,(FLD(I,J,M),J=1,3))
      WRITE(6,111) (M,(FLD(I,J,M),J=1,3),I=2,3)
      WRITE(6,115)
      WRITE(6,127) (M,(FLJ(I,J,M),J=1,3))
      WRITE(6,112) (M,(FLJ(I,J,M),J=1,3),I=2,3)
      WRITE(6,115)
16  CONTINUE
      WRITE (6,105)
C
C      RESULTANT COUPLED MODE DEPENDENT PARAMETERS
C
      DO 14 NN=1,NM
      DO 14 MM=1,NM
C
      DO 15 J=1,NJ
      CALL SUEOP(PHIT(1,J,NN),PHIT(1,J,MM),JMASS(J),TEM1)
      CALL DYADD(FCF(1,1,MM,NN),TEM1,FCF(1,1,MM,NN))
C
      CALL DYDOTV(JINERT(1,1,J),PHIR(1,J,MM),TEM)
      CALL VECROS(TEM,PHIR(1,J,NN),TFM)
      HF = .5
      CALL SCLV(HF,TFM,TEM)
      CALL VECROS(PHIR(1,J,MM),PHIR(1,1,NN),TFM)
      CALL SCLV(JMASS(J),TFM,TFM)
      CALL VECADD(TEM,TFM,TEM)
      CALL VECADD(FCK(1,MM,NN),TEM,FCK(1,MM,NN))
15  CONTINUE
      WRITE (6,117) MM,NN
      WRITE(6,115)
      WRITE(6,128) (MM,NN,(FCF(1,J,MM,NN),J=1,3))
      WRITE(6,113) (MM,NN,(FCF(1,J,MM,NN),J=1,3),I=2,3)
      WRITE(6,115)
      WRITE(6,114) (MM,NN,(FCK(1,MM,NN),I=1,3))
14  CONTINUE

```

```

C
100 FORMAT (1X,' ** ERROR ** CANNOT FIND HINGE POINT',I5,' IN GRID P
    *OINT TABLE, RETURN OUT OF SUBROUTINE NBOD2')
101 FORMAT (////,' GRID POINT LABELING, LOCATION AND MASS TABLES',/)
102 FORMAT (' IGRID(',I4,') =',I5,9X,'JLOC(',I4,') =',3E15.7,9X,
    *      'JMASS(',I4,') =',E15.7)
5 CONTINUE
  A = 1.0/TOTMAS
  CALL SCLV(A,CMLOC,CMLOC)
C
  DO 5 N=1,NJ
    TOTMAS = TOTMAS + JMASS(N)
    CALL SCLV(JMASS(N),JLOC(1,N),TEM)
    CALL VECADD(CMLOC,TEM,CMLOC)
8  Y(I) = X(I+1)
  NY = 3
  NR = IY(3)
  RETURN
7  IF(IFORM.NE.307) GO TO 9
  IY(1) = NR+1
  Y(1) = X(1)
  NY = 1
  NR = X(2)*FAC
  IF(NR.EQ.0) RETURN
  IY(2) = NR
  Y(2) = X(3)
  NY = 2
  NR = X(4)*FAC
  IF(NR.EQ.0) RETURN
  NR = NR-1
  RETURN
9  IF(IFORM.NE.308) GO TO 10
  NR = X(1)*FAC
  IF(NR.EQ.0) GO TO 14
  DO 11 I=1,3
    IY(I) = NR-1+I
11  Y(I) = X(I+1)
  NY = 3
  NR = IY(3)
  RETURN
10 IF(IFORM.NE.309) GO TO 12
  NR = X(1)*FAC
  IF(NR.EQ.0) GO TO 14
  DO 13 I=1,2
    IY(I) = NR-1+I
13  Y(I) = X(I+1)
  NY = 2
  NR = X(4)*FAC
  IF(NR.EQ.0) RETURN
  NR = NR-1
  RETURN
103 FORMAT (' XMAS =',E15.7,30X,' (TOTAL MASS OF BODY)')
104 FORMAT (////,' MODE SHAPE TABLE (INTERNAL NUMBERING) ',/)
105 FORMAT (////)
106 FORMAT (' PHIT(',I4,',' ,I2,') =',3E15.7,5X,
    *      ' PHIR(',I4,',' ,I2,') =',3E15.7)
107 FORMAT (////,' WARNING MASS MOMENT AT GRID POINT NON-ZERO, THAT IS
    * SMM ',I4,') =', 3E15.7,/, ' THIS EFFECT IGNORED BY N-BOD2. N-BOD2
    * ASSUMES EACH GRID POINT AT ELEMENT MASS CENTER ')

```



```

108 FORMAT (' FLA(',I2,') = ',3E15.7,' (COEFFICIENT TO FIND CENTER OF
* MASS LOCATION AFTER DEFORMATION)')
109 FORMAT (' FLB(',I2,') = ',3E15.7,' (COEFFICIENT TO FIND ANGULAR M
* OMENTUM DUE TO DEFORMATION)')
110 FORMAT (' FLC(',I2,') = ',3E15.7,' (COEFFICIENT TO FIND ANGULAR M
* OMENTUM DUE TO DEFORMATION)')
111 FORMAT (' FLD(',I2,') = ',3E15.7)
112 FORMAT (' FLJ(',I2,') = ',3E15.7)
113 FORMAT (' FCF(',I2,',',I2,') = ',3E15.7)
114 FORMAT (' FCK(',I2,',',I2,') = ',3E15.7,' (COEFFICIENT FOR CORIOLI
* S TORQUE DUE TO DEFORMATION)')
115 FORMAT (' ')
116 FORMAT (/ ,3X, 'FOR MODE',I3,' THE RESULTANT UNCOUPLED MODE DEPENDEN
* T PARAMETERS REQUIRED FOR N-BOD2 INPUT ARE')
117 FORMAT (/ ,3X, 'FOR MODES',I3,' AND',I3,' THE RESULTANT MODE DEPENDEN
* NT CROSS COUPLING PARAMETERS REQUIRED FOR N-BOD2 INPUT ARE')
118 FORMAT ('1',5X, '***** RESULTANT FLEXIBLE BODY DATA REQUIRED FOR IN
* PUT TO N-BOD2 *****',///)
119 FORMAT (' ',3E15.7,' (UNDEFORMED INERTIA TENSOR OF THE')
120 FORMAT (' X1 =',3E15.7,' FLEXIBLE BODY IN BODY COORDINATES')
121 FORMAT (' ',3E15.7,' RELATIVE TO BODY CENTER OF MASS')
122 FORMAT (' CA =',3E15.7,' (CENTER OF MASS VECTOR, HINGE POINT T
* O CM)')
123 FORMAT (' FLOM(',I2,') =',E15.7,' RAD/SEC (MODAL FREQUENCY)')
124 FORMAT (' ZETA(',I2,') =',E15.7,' (MODAL DAMPING ZETA)')
125 FORMAT (' CB = VECTOR IN CONTIGUOUS BODY TO HINGE POINT DEFINED
* AT INTERNAL GRID POINT',I5,' (EXTERNAL GRID POINT',I5,' IN BULK D
* ATA)')
126 FORMAT (' FLD(',I2,') = ',3E15.7,' (COEFFICIENT TO FIND INERTIA T
* ENSOR AFTER DEFORMATION)')
127 FORMAT (' FLJ(',I2,') = ',3E15.7,' (COEFFICIENT FOR CENTRIPITAL T
* ORQUE DUE TO DEFORMATION)')
128 FORMAT (' FCF(',I2,',',I2,') = ',3E15.7,' (COEFFICIENT FOR CENTRIP
* ITAL TORQUE DUE TO DEFORMATION)')
RETURN
END

```

SUBROUTINE INTERP(INPT,IMAT)

```

C
C   THIS ROUTINE SETS UP A CODE SEQUENCE WHICH WILL BE USED
C   TO READ MIXED INTEGER AND FLOATING POINT DATA FROM
C   THE DMI CARDS
C   BLANK FIELD ON LAST CARD WILL BE INTERPRETED AS AN INTEGER
C   BUT READ LATER IN AS ZERO, A CHECK FOR WHICH IS MADE
C
C   INTEGER INPT(80)
C   DATA IDOT,IEE/1H.,1HE/
C   IF(INPT(12).EQ.IDOT.AND.INPT(21).EQ.IEE) GO TO 1
C   IF(INPT(44).EQ.IDOT.AND.INPT(53).EQ.IEE) GO TO 2
C   IMAT = 310
C
C   310 - FORMAT  I,E,I,E
C
C   RETURN
C   2 IF(INPT(60).EQ.IDOT.AND.INPT(69).EQ.IEE) GO TO 3
C   IMAT = 309
C
C   309 - FORMAT  I,E,E,I

```

```

      RETURN
3 IMAT = 308
C          308 - FORMAT  I,E,E,E
      RETURN
1 IF(INPT(28).EQ.IDOT.AND.INPT(37).EQ.IEE) GO TO 4
  IF(INPT(60).EQ.IDOT.AND.INPT(69).EQ.IEE) GO TO 5
  IMAT = 307
C          307 - FORMAT  E,I,E,I  OR  E
      RETURN
5 IMAT = 306
C          306 - FORMAT  E,I,E,E
      RETURN
4 IF(INPT(44).EQ.IDOT.AND.INPT(53).EQ.IEE) GO TO 6
  IMAT = 305
C          305 - FORMAT  E,E,I,E  OR  E,E
      RETURN
6 IF(INPT(60).EQ.IDOT.AND.INPT(69).EQ.IEE) GO TO 7
  IMAT = 304
C          304 - FORMAT  E,E,E,I  OR  E,E,E
      RETURN
7 IMAT = 303
C          303 - FORMAT  E,E,E,E
      RETURN
END

```

SUBROUTINE DECODE(IFORM,X,NR,Y,IY,NY)

```

C
C***      ERROR IN TERMINOLOGY ELEMENTS COMING IN BY ROWS,
C          NR AND IY(I) REFER TO COLUMN NUMBER
C
C          OUTPUT OF DECODE
C          NY      = NUMBER OF REAL NUMBERS ON DMI CARD
C          IY(I)   = ROW LOCATION FOR I-TH REAL NUMBER ON DMI CARD
C          Y(I)    = I-TH REAL NUMBER
C          NR      = NEXT REAL NUMBER TO BE PUT IN ROW NR+1 UNLESS
C                   FIELD 1 OF NEXT CARD CONTAINS AN INTEGER
C

```

```

      REAL*8 X(4), Y(4)
      INTEGER IY(4)
      FAC = 1.00001D+08
      IF(IFORM.NE.303) GO TO 1
      DO 2 I=1,4
        IY(I) = NR+I
2 Y(I) = X(I)
      NY = 4
      NR = IY(4)
      RETURN
1 IF(IFORM.NE.304) GO TO 3
      DO 4 I=1,3
        IY(I) = NR+I
4 Y(I) = X(I)
      NY = 3
      NR = X(4)*FAC-1
      RETURN
3 IF(IFORM.NE.305) GO TO 5

```

```

      DO 6 I=1,2
      IY(I) = NR+I
6     Y(I) = X(I)
      NY = 2
      NR = X(3)*FAC
      IF(NR.EQ.0) RETURN
      IY(3) = NR
      Y(3) = X(4)
      NY = 3
      RETURN
5     IF(IFORM.NE.306) GO TO 7
      IY(1) = NR+1
      Y(1) = X(1)
      NY = 1
      NR = X(2)*FAC
      IF(NR.EQ.0) RETURN
      DO 8 I=2,3
      IY(I) = NR-2+I
12    IF(IFORM.NE.310) GO TO 14
      NR = X(1)*FAC
      IF(NR.EQ.0) GO TO 14
      IY(1) = NR
      Y(1) = X(2)
      NR = X(3)*FAC
      IF(NR.EQ.0) RETURN
      IY(2) = NR
      Y(2) = X(4)
      NY = 2
      RETURN
14    WRITE(6,100) IFORM
100   FORMAT (' IFORM =',I5,' ERROR')
      STOP
      END

```

```

      SUBROUTINE INTAPE (NTAPE,TAPEID)
      REAL*8 TAPEID,BUF,EOT
      DATA IZ1,BUF,EOT/1,0.0,3HEOT/
      DATA NOT / 6/

C
C  INITIALIZE TAPE FOR SUBROUTINE WTAPE.
C  CALLS FORMA SUBROUTINE PAGEHD.
C  CODED BY RF HRUDA. JULY 1968.
C  REVISED BY R A PHILIPPUS. APRIL 1969.
C
C  SUBROUTINE ARGUMENTS (ALL INPUT)
C  NTAPE = NUMBER OF TAPE. (E.G. 10).
C  TAPEID = TAPE IDENTIFICATION. (E.G. T1234). (A6 FORMAT).
C
2001  FORMAT (//// 14H LOGICAL UNIT IZ, 7H, TAPE A6,
      *          23H, HAS BEEN INITIALIZED.)
C
      REWIND NTAPE
      WRITE (NTAPE) TAPEID,IZ1,EOT,(BUF,I=1,16)
      REWIND NTAPE

```

WRITE (NOT,2001) NTAPE,TAPEID

RETURN
END

SUBROUTINE WTAPE (A,NRA,NCA,ANAME,KR,NTAPE)
REAL*8 A,ANAME,IRUNNO,DATE,
* BUF,EOT,DENSE,TAPEID,IEOTCK
DIMENSION A(KR,1)
DATA IRUNNO,DATE/6HPREPRS,6HDATE /
DATA BUF,EOT,DENSE/0.D 0,3HEOT,5HDENSE/
DATA NOT / 6/

WRITE MATRIX A ON TAPE.

INITIALIZE TAPE WITH SUBROUTINE INTAPE.

REWIND TAPE BEFORE FIRST USE OF THIS SUBROUTINE.

NOTE...THIS ROUTINE IS DESIGNED SPECIFICALLY FOR WRITING ON A DISK

(EG CDC-6400 DISK). USING THIS ROUTINE TO WRITE ON A PHYSICAL
TAPE DIRECTLY (IE WITHOUT USING THE DISK AS AN INTERMEDIARY)
WILL PROBABLY GIVE POOR RESULTS (DUE TO THE TOLERANCE
CHARACTERISTICS OF A TAPE DRIVE) AND SHOULD BE AVOIDED IF AT
ALL POSSIBLE.

...THE CDC-6400 DISK IS AUTOMATICALLY ENDFILED AFTER EACH WRITE.

CODED BY W A BENFIELD. MARCH 1966.

REVISED BY R A PHILIPPUS. APRIL 1969.

REVISED BY RF HRUDA. NOVEMBER 1970.

SUBROUTINE ARGUMENTS (ALL INPUT)

A = MATRIX TO BE WRITTEN ON TAPE. SIZE(NRA,NCA).

NRA = NUMBER OF ROWS OF MATRIX A.

NCA = NUMBER OF COLS OF MATRIX A.

ANAME = MATRIX IDENTIFICATION. (A6 FORMAT).

KR = ROW DIMENSION OF A IN CALLING PROGRAM.

NTAPE = NUMBER OF TAPE. (E.G. 10).

INTERNAL VARIABLES THAT ARE PUT ON TAPE (TRANSFERRED THRU COMMON).

IRUNNO IS RUN NUMBER OF PROBLEM. (A6 FORMAT).

DATE IS DATE. (A6 FORMAT). FOR EXAMPLE 15FE65.

IF (NRA .LT. 1 .OR. NCA .LT. 1) GO TO 999

SEARCH TAPE FOR END OF WRITTEN DATA.

10 READ (NTAPE) TAPEID,LN,IEOTCK
IF (IEOTCK .EQ. EOT) GO TO 20
READ (NTAPE)
GO TO 10

END OF WRITTEN DATA HAS BEEN FOUND.

20 BACKSPACE NTAPE

WRITE (NTAPE) TAPEID,LN,BUF,IRUNNO,ANAME,NRA,NCA,DATE,DENSE,

* (BUF,I=1,10)

WRITE (NTAPE) ((A(I,J),I=1,NRA),J=1,NCA)

LN = LN + 1

WRITE (NTAPE) TAPEID,LN,EOT,(BUF,I=1,16)

```

BACKSPACE NTAPE
RETURN
C
  999 WRITE (NOT,1000)
1000 FORMAT (1H1,43HERROR IN SUBROUTINE WTAPE, PROGRAM STOPPED.)
      STOP
      END

SUBROUTINE LTAPE (NTAPE)
  REAL*8 TAPEID,IRUNNO,ANAME,IEOTCK,DATE,ITYPE,ICLK,EOT,
  *      DENSE,SPARSE,SPART
      DATA NOT / 6/
      DATA EOT,DENSE,SPARSE,SPART/ 3HEOT,5HDENSE,6HSPARSE,5HSPART/
C
C  LIST HEADINGS OF MATRICES ON TAPE.
C  CALLS FORMA SUBROUTINE PAGEHD.
C  CODED BY RF HRUDA. JULY 1968.  REVISED NOVEMBER 1970.
C  REVISED BY R A PHILIPPUS. APRIL 1969.
C
C  SUBROUTINE ARGUMENTS (ALL INPUT)
C  NTAPE = NUMBER OF TAPE. (E.G. 10).
C
2001 FORMAT (//36X35HLISTING OF MATRICES ON LOGICAL UNIT13,7H, TAPE A6)
2002 FORMAT (//30X35HLISTING OF MATRICES ON LOGICAL UNIT13,7H, TAPE A6,
  *      12H (CONTINUED))
2003 FORMAT (27X69(1H-)/27X3HNO.3X7HRUN NO.4X4HNAME5X5HNROWS4X5HNCOLS4X
  *      4HDATE6X3HNN23X9HPARTITION/
  *      27X3H---3X6H-----4X6H-----4X5H-----
  *      4X5H-----3X6H-----5X3H---3X9H----- / )
2004 FORMAT (25XI5,3XA6,4XA6,3XI5,4XI5,4XA6,3XI5,3XI4,1H/I4)
2005 FORMAT (/27X12HEND OF LIST.)
C
      REWIND NTAPE
      READ (NTAPE) TAPEID
      REWIND NTAPE
      L=0
C
12 CONTINUE
  IF(L .EQ. 0) WRITE (NOT,2001) NTAPE,TAPEID
  IF(L .NE. 0) WRITE (NOT,2002) NTAPE,TAPEID
  WRITE (NOT,2003)
  NLINE=1
13 L=L+1
  READ (NTAPE) TAPEID,LN,IEOTCK,IRUNNO,ANAME,NR,NC,DATE,ITYPE,NNZ,
  *      NP,NPT
  IF (L .EQ. 1) ICHK = IRUNNO
  IF (ICLK .EQ. IRUNNO) GO TO 15
  NLINE=NLINE+1
  WRITE (NOT,2004)
  ICHK = IRUNNO
15 IF (IEOTCK .EQ. EOT) GO TO 30
  READ (NTAPE)
  IF (ITYPE .EQ. DENSE ) WRITE (NOT,2004)
  *      LN,IRUNNO,ANAME,NR,NC,DATE

```

```

      IF (ITYPE .EQ. DENSE ) GO TO 20
      IF (ITYPE .EQ. SPARSE) WRITE (NOT,2004)
      * LN,IRUNNO,ANAME,NR,NC,DATE,NNZ
      IF (ITYPE .EQ. SPARSE) GO TO 20
      IF (ITYPE .EQ. SPART ) WRITE (NOT,2004)
      * LN,IRUNNO,ANAME,NR,NC,DATE,NNZ,NP,NPT
      IF (ITYPE .EQ. SPART ) GO TO 20
      WRITE (NOT,2004) LN,IRUNNO,ANAME,NR,NC,ITYPE
20  NLINE=NLINE+1
      IF(NLINE.GT.43) GO TO 12
      GO TO 13

```

```

C
30  WRITE (NOT,2004) LN,IEOTCK
      WRITE (NOT,2005)
      REWIND NTAPE
      RETURN
      END

```

```

      SUBROUTINE RTAPE (IARUNO,IANAME, A,NRA,NCA, KR,KC,NTAPE)
      REAL*8 A,IARUNO,IANAME,TAPEID,IEOTCK,ITRUNO,ITNAME,
      * DATE,ITYPE,DENSE,EOT
      DIMENSION A(KR,1)
      DATA NOT / 6 /
      DATA DENSE,EOT / 5HDENSE,3HEOT /

```

```

C
C READ MATRIX A FROM TAPE BY IDENTIFICATION OF IARUNO,IANAME.
C CALLS FORMA SUBROUTINES LTAPE,PAGEHD,ZZHOMB.
C CODED BY WA BENFIELD. JUNE 1966.
C LAST REVISION BY R F HRUDA. SEPTEMBER 1971.
C
C SUBROUTINE ARGUMENTS
C IARUNO = INPUT RUN NUMBER OF MATRIX A. (A6 FORMAT).
C IANAME = INPUT MATRIX IDENTIFICATION. (A6 FORMAT).
C A = OUTPUT MATRIX READ FROM TAPE. SIZE(NRA,NCA).
C NRA = OUTPUT NUMBER OF ROWS OF MATRIX A. WILL BE READ FROM TAPE.
C NCA = OUTPUT NUMBER OF COLS OF MATRIX A. WILL BE READ FROM TAPE.
C KR = INPUT ROW DIMENSION OF A IN CALLING PROGRAM.
C KC = INPUT COL DIMENSION OF A IN CALLING PROGRAM.
C NTAPE = INPUT NUMBER OF TAPE. (E.G. 10).
C

```

```

3001 FORMAT (29H1RTAPE CANNOT FIND RUNNO = A6 /
      * 21X 8HANAME = A6 / 29X 6H----- )

```

```

C
      NTIME = 0
C SEARCH TAPE FOR CORRECT HEADING.
      5 READ (NTAPE) TAPEID,LN,IEOTCK,ITRUNO,ITNAME,NRA,NCA,DATE,ITYPE,NNZ
      IF (ITRUNO .EQ. IARUNO .AND. ITNAME .EQ. IANAME) GO TO 10
      IF (IEOTCK .EQ. EOT) GO TO 20
      READ (NTAPE)
      GO TO 5

```

```

C
C MATRIX HAS BEEN FOUND.
10

```

```

                                NERROR=1
        IF (ITYPE .NE. DENSE .AND. NNZ .NE. 0) GO TO 999
                                NERROR=2
        IF (NRA.GT.KR .OR. NCA.GT.KC) GO TO 999
        READ (NTAPE) ((A(I,J),I=1,NRA),J=1,NCA)
        RETURN
C
C MATRIX CANNOT BE FOUND. SEARCH TAPE ONCE MORE.
20 NTIME = NTIME+1
                                NERROR=3
        IF (NTIME .EQ. 2) GO TO 998
        REWIND NTAPE
        GO TO 5
C
998 WRITE (NOT,3001) IARUNO,IANAME
999 CALL LTAPE (NTAPE)
        WRITE (NOT,3002) NERROR
3002 FORMAT (1H1,43HERROR IN SUBROUTINE RTAPE, PROGRAM STOPPED,
*          10H NERROR = ,I3)
        STOP
        END

        SUBROUTINE WRITE (A,NR,NC,ANAME,KR)
        REAL*8 A,ANAME
        DIMENSION A(KR,1)
        DATA NOT / 6/
C
C WRITE MATRIX OF REAL NUMBERS ON PAPER.
C REQUIRES 123 COLUMN (MINIMUM) PRINTER.
C UP TO 10 DATA FIELDS PER LINE. PRINTS ONLY NON-ZERO FIELD ROWS.
C CALLS FORMA SUBROUTINE PAGEHD.
C CODED BY RL WOHLN. DECEMBER 1968.
C
C SUBROUTINE ARGUMENTS (ALL INPUT)
C A = MATRIX TO BE PRINTED. SIZE(NR,NC).
C NR = NUMBER OF ROWS IN MATRIX A.
C NC = NUMBER OF COLS IN MATRIX A.
C ANAME = MATRIX IDENTIFICATION. (A6 FORMAT).
C KR = ROW DIMENSION OF A IN CALLING PROGRAM.
C
.2010 FORMAT (//15H OUTPUT MATRIX A6,2X 1H(14,2H X 14,2H ) //
*          10X,10(7X,1H( 12,1H))//)
2020 FORMAT (//15H OUTPUT MATRIX A6,2X 1H(14,2H X 14,2H )
*          3X, 9HCONTINUED //10X,10(7X,1H( 12,1H))//)
2030 FORMAT (1X,215,2X,1P10D11.3)
2040 FORMAT (14HOEND OF WRITE.)
C
C PULL UP A NEW PAGE FOR MATRIX AND PRINT MATRIX NAME.
        WRITE (NOT,2010) ANAME,NR,NC,(L,L=1,10)
        NLINE = 0
C
        DO 60 I=1,NR
        NZERO = 0
        JS = 1
10 JE = JS+9

```

```

      IF (JE .GT. NC) JE=NC
C   SEE IF ELEMENTS ARE ZERO.
      DO 20 J=JS,JE
      IF (A(I,J) .NE. 0.0) GO TO 30
20  CONTINUE
      GO TO 40
30  NLINE = NLINE+1
      IF (NLINE .LE. 44) GO TO 35
      WRITE (NOT,2020) ANAME,NR,NC,(L,L=1,10)
      NLINE = 1
35  WRITE (NOT,2030) I,JS,(A(I,J), J=JS,JE)
      NZERO = 1
40  IF (JE .EQ. NC) GO TO 50
      JS = JS+10
      GO TO 10
C   SKIP A SPACE BETWEEN EACH ROW IF THERE ARE MORE THAN 10 COLUMNS
C   AND SOMETHING HAS BEEN WRITTEN.
50  IF (NC.LE.10 .OR. NZERO.EQ.0 .OR. I.EQ.NR) GO TO 60
      NLINE = NLINE+1
      WRITE (NOT,2030)
60  CONTINUE
C
      WRITE (NOT,2040)
      RETURN
      END

```

```

      SUBROUTINE VECADD(V1,V2,S)
C   ADDS VECTOR V1 TO V2 RESULT IN S
      IMPLICIT REAL*8(A-H,O-Z,$)
      DIMENSION V1(3),V2(3), S(3)
      S(1) = V1(1) + V2(1)
      S(2) = V1(2) + V2(2)
      S(3) = V1(3) + V2(3)
      RETURN
      END

```

```

      SUBROUTINE VECSUB(V1,V2,D)
C   SUBTRACTS VECTORS V1-V2=D
      IMPLICIT REAL*8(A-H,O-Z,$)
      DIMENSION V1(3),V2(3),D(3)
      D(1) = V1(1) - V2(1)
      D(2) = V1(2) - V2(2)
      D(3) = V1(3) - V2(3)
      RETURN
      END

```



```

C      SUBROUTINE SCLV(SC,V,P)
      SCALAR * VECTOR
      IMPLICIT REAL*8(A-H,O-Z,$)
      DIMENSION V(3),P(3)
      P(1) = SC*V(1)
      P(2) = SC*V(2)
      P(3) = SC*V(3)
      RETURN
      END

```

```

C      SUBROUTINE VECDOT(V1,V2,D)
      VECTOR DOT PRODUCT
      IMPLICIT REAL*8(A-H,O-Z,$)
      DIMENSION V1(3),V2(3)
      D = V1(1)*V2(1) + V1(2)*V2(2) + V1(3)*V2(3)
      RETURN
      END

```

```

C      SUBROUTINE VECROS (V1,V2,C)
      VECTOR CROSS PRODUCT C = V1 X V2
      IMPLICIT REAL*8(A-H,O-Z,$)
      DIMENSION V1(3),V2(3),C(3)
      C(1) = V1(2)*V2(3)- V1(3)*V2(2)
      C(2) = V1(3)*V2(1)- V1(1)*V2(3)
      C(3) = V1(1)*V2(2)- V1(2)*V2(1)
      RETURN
      END

```

```

C      SUBROUTINE TRIPVP(V1,V2,V)
C      COMPUTES STANDARD VECTOR TRIPLE PRODUCT
C
C      V = V1X(V1XV2)
C      = V1*(V1.V2) - V2*(V1.V1)
C
C      IMPLICIT REAL*8(A-H,O-Z,$)
C      DIMENSION V1(3),V2(3),V(3)
C      A = V1(1)*V2(1) + V1(2)*V2(2) + V1(3)*V2(3)
C      B = V1(1)*V1(1) + V1(2)*V1(2) + V1(3)*V1(3)
C      V(1) = V1(1)*A - V2(1)*B
C      V(2) = V1(2)*A - V2(2)*B
C      V(3) = V1(3)*A - V2(3)*B
C      RETURN
C      END

```

```

SUBROUTINE DYADD(D1,D2,D)
C   ADDS TWO DYADS
C       D = D1 + D2
IMPLICIT REAL*8(A-H,O-Z,$)
DIMENSION D1(3,3), D2(3,3), D(3,3)
DO 1 I=1,3
DO 1 J=1,3
1 D(I,J) = D1(I,J) + D2(I,J)
RETURN
END

```

```

SUBROUTINE SCLD(A,D,T)
IMPLICIT REAL*8(A-H,O-Z,$)
DIMENSION D(3,3),T(3,3)
C   MULTIPLY SCALAR BY A TENSOR
T(1,1) = A*D(1,1)
T(2,1) = A*D(2,1)
T(3,1) = A*D(3,1)
T(1,2) = A*D(1,2)
T(2,2) = A*D(2,2)
T(3,2) = A*D(3,2)
T(1,3) = A*D(1,3)
T(2,3) = A*D(2,3)
T(3,3) = A*D(3,3)
RETURN
END

```

```

SUBROUTINE DYDOTV(A,V,D)
C   SCALAR DOT PRODUCT OF DYAD AND VECTOR
C       D = A.V
IMPLICIT REAL*8(A-H,O-Z,$)
DIMENSION D(3),A(3,3),V(3)
DO 1 I=1,3
D(I) = 0
DO 1 J=1,3
1 D(I) = D(I) + A(I,J)*V(J)
RETURN
END

```

```

SUBROUTINE VXDYOV(V1,DY,V)
C   COMPUTES VECTOR X (DYAD . VECTOR)
C       V = V1 X (DY . V1)
IMPLICIT REAL*8(A-H,O-Z,$)
DIMENSION V1(3),V2(3),V(3),DY(3,3)
DO 1 K=1,3

```

```

V2(K) = 0.D0
DO 1 J=1,3
1 V2(K) = V2(K) + DY(K,J)*V1(J)
V(1) = V1(2)*V2(3) - V1(3)*V2(2)
V(2) = V1(3)*V2(1) - V1(1)*V2(3)
V(3) = V1(1)*V2(2) - V1(2)*V2(1)
RETURN
END

```

```

SUBROUTINE DYT0V (D,X1,X)
C   USE TO TAKE SCALAR DOT PRODUCT OF TRANSPOSE OF
C   TENSOR D WITH VECTOR X1
C   NEEDED SINCE TENSORS IN SYMMERIC MATRIX OF INERTIA TENSORS ARE IN
C   NON SYMMETRIC
IMPLICIT REAL*8(A-H,O-Z,$)
DIMENSION D(3,3),X1(3),X(3)
DO 1 I=1,3
X(I) = 0
DO 1 J=1,3
X(I) = X(I) + D(J,I)*X1(J)
1 CONTINUE
RETURN
END

```

```

SUBROUTINE VODYOV(V1,DY,V2,X)
C   COMPUTES THE SCALAR TRIPLE PRODUCT
C   VECTOR . (DYAD . VECTOR)
C   V1.(DY.V2)
IMPLICIT REAL*8(A-H,O-Z,$)
DIMENSION V1(3),DY(3,3),V2(3),TEM(3)
DO 1 K=1,3
TEM(K) = 0.D0
DO 1 J=1,3
1 TEM(K) = TEM(K) + DY(K,J)*V2(J)
X = 0
DO 2 J=1,3
2 X = X + V1(J)*TEM(J)
RETURN
END

```

```

SUBROUTINE DYOP(V,D)
C   TRANSFORMS VECTOR V1 INTO SKEW DYAD
IMPLICIT REAL*8(A-H,O-Z,$)
DIMENSION V(3),D(3,3)
D(1,1) = 0
D(1,2) = V(3)

```

```

D(1,3) = -V(2)
D(2,1) = -V(3)
D(2,2) = 0
D(2,3) = V(1)
D(3,1) = V(2)
D(3,2) = -V(1)
D(3,3) = 0
RETURN
END

```

```

SUBROUTINE SUEOP(V1,V2,XM,D)
IMPLICIT REAL*8(A-H,O-Z,$)
DIMENSION V1(3),V2(3),D(3,3)
C   USED TO COMPUTE THE PSUEDO INERTIA TENSOR
C   OF BODY LAMBA WITH RESPECT TO THE ORIGIN OF NEST K-1 AND
C   THE HINGE POINT I-1 WHICH IS ON THE TOPOLOGICAL PATH FROM
C   BODY 1 TO BODY LAMBA
C       BLOCK G SUPPER GAMBA,SUB K-1,I-1 EQUATION 2-55 OF X-732-71-70
C           D = XM*((V1.V2)*1 - V2 V1)
C
C   XM - SCALAR
C   V1 - VECTOR
C   V2 - VECTOR
C   1 - UNIT DYAD
C   * - SCALAR MULTIPLICATION
C   . - VECTOR SCALAR MULTIPLICATION
C   BLANK - TENSOR MULTIPLICATION
C   NOTE THAT IN GENERAL THE PSUEDO INERTIA TENSOR IS NON SYMMETRIC
C
D(1,1) = XM*(V1(2)*V2(2) + V1(3)*V2(3))
D(1,2) = -XM*V2(1)*V1(2)
D(1,3) = -XM*V2(1)*V1(3)
D(2,1) = -XM*V2(2)*V1(1)
D(2,2) = XM*(V2(1)*V1(1) + V2(3)*V1(3))
D(2,3) = -XM*V2(2)*V1(3)
D(3,1) = -XM*V2(3)*V1(1)
D(3,2) = -XM*V2(3)*V1(2)
D(3,3) = XM*(V2(1)*V1(1) + V2(2)*V1(2))
RETURN
END

```

```

SUBROUTINE VECXDY(P,T,D)
REAL*8 P(3), T(3,3), D(3,3)
C
C   COMPUTES VECTOR CROSS DYAD
C       D = P X T
C
D(1,1) = P(2)*T(3,1) - P(3)*T(2,1)
D(1,2) = P(2)*T(3,2) - P(3)*T(2,2)
D(1,3) = P(2)*T(3,3) - P(3)*T(2,3)
D(2,1) = P(3)*T(1,1) - P(1)*T(3,1)

```

```
D(2,2) = P(3)*T(1,2) - P(1)*T(3,2)
D(2,3) = P(3)*T(1,3) - P(1)*T(3,3)
D(3,1) = P(1)*T(2,1) - P(2)*T(1,1)
D(3,2) = P(1)*T(2,2) - P(2)*T(1,2)
D(3,3) = P(1)*T(2,3) - P(2)*T(1,3)
RETURN
END
```

Goddard Space Flight Center
National Aeronautics and Space Administration
Greenbelt, Maryland October 1977

APPENDIX A

INPUT/OUTPUT SUBROUTINE EXPLANATIONS

APPENDIX A

INPUT/OUTPUT SUBROUTINE EXPLANATIONS

This appendix presents excerpts from *Synthesis of Dynamic Systems Using FORMA-Fortran Matrix Analysis*, MCR-71-75, Martin Marietta Corporation, Denver, Colorado, May 1971, that explain the subroutines from the FORMA library used in the digital computer program.

COMENT

Subroutine COMENT reads input comment cards and reproduces each card in the printed output of the computer run. Each comment card may have any keypunch symbol in card columns 1 through 78. A use of COMENT is to print an explanation of coordinates used in a computer run. Thus, this information is always retained with a run to correlate matrix location numbers with physical coordinates.

INTAPE

Subroutine INTAPE initializes a tape* for the FORMA tape system by writing EOT (end of tape) at the beginning of the tape (disk). All FORMA tape subroutines recognize the EOT as the end of written data. Each "new" tape (disk) must be initialized with this subroutine INTAPE to make the tape (disk) compatible with the other FORMA tape subroutines (LTAPE, RTAPE, WTAPE, and UPDATE).

A "new" tape (disk) is defined as a tape (disk) for which it is desired to begin writing matrix data at the front of the tape (disk). Thus, a "new" tape (disk) could be one that contains obsolete FORMA matrix data, as well as one on which the FORMA system has never written.

As an example, pertinent statements from a program containing INTAPE could be:

```
DATA NIT,NOT/5,6/
1001 FORMAT (12A6)
NRTAPE = 10
READ (NIT, 1001) IFINIT, TAPEID
IF (IFINIT .EQ. 6HINITIL) CALL INTAPE(NRTAPE,TAPEID)
.
.
.
```

*A disk is preferred. See writeup of subroutine WTAPE at the end of this appendix.

The input data (beginning in card column 1) to this example program would be either of the following:

- INITILTXXXX, if the tape is to be initialized. (TXXXX represents the particular tape number used; (e.g., T1234).
- NOINIT, if the tape is *not* to be initialized. The tape identification is not needed.

LTAPE

Subroutine LTAPE lists the matrix headings (see subroutine WTAPE writeup) written on a FORMA tape (or disk). The following matrix headings were written by subroutine WTAPE and consist of:

NO.	= Matrix number on tape
RUN NO.	= Run number of problem when matrix was written on tape
NAME	= Matrix name
NROWS	= Number of rows of matrix
NCOLS	= Number of columns of matrix
DATE	= Date when matrix was written on tape
NNZ	= Number of nonzeros (used only in sparse FORMA in which only nonzeros are used)
PARTITION	= Partition number of sparse matrix

READ

Subroutine READ reads a matrix of real numbers (a FORTRAN term for numbers with a decimal point) from either cards or tape into the computer. The matrix is then printed so that these input data are recorded with the answers of a run. A print suppression option is available for a matrix read from tape. On option, the matrix read from either cards or tape may be written on a tape (by subroutine WTAPE).

The first data card read by subroutine READ contains the information for indicating whether cards or tape will be used. The information entered on this card (and subsequent cards for card input) is as follows:

Card Data Input Form

In the following, an asterisk denotes required entries. Other entries are optional.

	<u>Card Columns</u>	<u>Format Type¹</u>	<u>Entry</u>
First Card	1-6	A	*Matrix name—will appear in printout
	7-10	I	*Matrix row size
	11-15	I	*Matrix column size
	16-69	A	Any remarks to further identify the input matrix
Write-Tape Options	72		\$.—Only if the Write-Tape is to be initialized by subroutine INTAPE. The Write-Tape identification will be from card columns 73 through 78.
	72	or	Anything other than \$ if the Write-Tape is not to be initialized.
	73-78	A	Write-Tape identification (e.g., T1234)—use with \$ in card column 72.
	73-78	or	REWIND—The Write-Tape will be rewound before being used.
	73-76	or	LIST—The Write-Tape will be listed by subroutine LTAPE after the matrix has been written on the Write-Tape.
	73-78	or	Anything else will be ignored.
	79-80	I	Write-Tape number (e.g., 21)
		or	Blank if the matrix is not to be written on tape.
Middle Cards	1-5	I	*Row number of matrix elements on card
	6-10	I	*Column number of matrix element in first data field.

	<u>Card Columns</u>	<u>Format Type¹</u>	<u>Entry</u>
Middle Cards (continued)	11-27	E	*First data field with matrix elements ²
	28-44	E	*Second data field with matrix elements ²
	45-61	E	*Third data field with matrix elements ²
	62-78	E	*Fourth data field with matrix elements ²
Last Card	1-10	I	*Ten zeroes

Note 1: Format Type A allows any keypunch symbol. Format Type I allows only integer numbers right-justified in the field. Format Type E allows only read numbers (a FORTRAN term for numbers with a decimal point) anywhere in the field).

Note 2: Only nonzero elements need be entered.

As an example of card input to subroutine READ, consider the following matrix:

$$[A1*C]_{3 \times 6} = \begin{bmatrix} 1. & 0. & 3. & 0. & 6. & 5. \\ 0. & 2. & 4. & 0. & 0. & 0. \\ 0. & 7. & 0. & 0. & 0. & 0. \end{bmatrix}$$

This matrix is also to be written on tape 21 that is to be initialized and identified as T4334. Figure A-1 demonstrates how this information could be written on a coding form to facilitate keypunching to cards.

Tape Data Input Form

In the following, an asterisk denotes required entries. Other entries are optional. Only one card is used for each matrix read.

	<u>Card Columns</u>	<u>Format Type¹</u>	<u>Entry</u>
One Card	1-6	A	*Name of matrix to be read from Read-Tape.
	10		Zero—Read-Tape will move forward from its present position and search to the end of the tape. If the matrix is not found on the first end-of-tape encounter, the tape will automatically rewind and make one more pass. If it is not found on the second end-of-tape encounter, an error message will be printed and the program will stop.
	7-10	I or	Minus the location number of matrix on the Read-Tape. Tape will be positioned at the beginning of the location specified and will then continue as described above for a zero in column 10.
	11-15	I	*Read-Tape number (e.g., 11)— If positive, the matrix read will be printed in the output. If negative, the matrix read will not be printed in the output.
	16-21	A	*Run number of matrix to be read from the Read-Tape.
	22-27		REWIND—Read-Tape will be rewound before being used.
	22-25	or	LIST—Read-Tape will be listed by subroutine LTAPE.

	<u>Card Columns</u>	<u>Format Type¹</u>	<u>Entry</u>
	22-27	or	Anything else will be considered as part of the remarks described below.
	28-69	A	Any remarks to further identify the input matrix.
Write-Tape Options	72		\$.—Only if the Write-Tape is to be initialized by subroutine INTAPE. The Write-Tape identification will be from card columns 73 through 78.
	72	or	Anything other than \$ if the Write-Tape is not to be initialized.
	73-78	A	Write-Tape identification (e.g., T1234)—Use with \$ in card column 72.
	73-78	or	REWIND—Write-Tape will be rewound before being used.
	73-76	or	LIST—Write-Tape will be listed by subroutine LTAPE after the matrix has been written on the Write-Tape.
	73-78	or	Anything else will be ignored.
	79-80	I or	Write-Tape number (e.g., 21) Blank if the matrix is not to be written on tape.

Note 1: Format Type A allows any keypunch symbol. Format Type I allows only integer numbers right-justified in the field.

As examples of tape input to subroutine READ, consider:

- Example 1—A matrix named AB2 with run number of RUN-46 is to be read from tape number 11 into the computer and printed. This matrix is also to be written on tape number 22 that is to be initialized and identified as T4321.

- Example 2—A matrix named XYZ4 with run number of TKD is on tape number 13 twice—the first time at location 29 and the second time at location 54. It is desired to read the second matrix.

Figure A-2 demonstrates how these two examples would be written on a coding form to facilitate keypunching to cards.

READIM

Subroutine READIM reads a matrix of integer numbers from either cards or tape into the computer. The matrix is then printed so that these input data are recorded with the answers of a run. A print suppression option is available for a matrix read from tape. On option, the matrix read from either cards or tape may be written on a tape (by subroutine WTAPE).

The first data card read by subroutine READIM contains the information to indicate whether cards or tape will be used. The information entered on this card (and subsequent cards for card input) is given below.

Card Data Input Form

Required entries are denoted by an asterisk below. Any other entry is optional.

	<u>Card Columns</u>	<u>Format Type¹</u>	<u>Entry</u>
First Card	1-6	A	*Matrix name—Will appear in printout
	7-10	I	*Matrix row size
	11-15	I	*Matrix column size
	16-69	A	Any remarks to further identify the input matrix
Write-Tape Options	72		\$—Only if the Write-Tape is to be initialized by subroutine INTAPE. The Write-Tape identification will be from card columns 73 through 78.
	72	or	Anything other than \$ if the Write-Tape is not to be initialized.
	73-78	A	Write-Tape identification (e.g., T1234)—Use with \$ card column 72.

	<u>Card Columns</u>	<u>Format Type¹</u>	<u>Entry</u>
	73-78	or	REWIND—Write-Tape will be rewound before being used.
	73-76	or	LIST—Write-Tape will be listed by subroutine LTape after the matrix has been written on the Write-Tape.
	73-78	or	Anything else will be ignored.
	79-80	I or	Write-Tape number (e.g., 21) Blank if the matrix is not to be written on tape.
Middle Cards	1-5	I	*Row number of matrix elements on card
	6-10	I	*Column number of matrix element in first data field
	11-15	I	*First data field with matrix elements ²
	16-20	I	*Second data field with matrix elements ²
	etc.		
	76-80	I	*Fourteenth data field with matrix elements ²
Last Card	1-10	I	*Ten zeroes

Note 1: Format Type A allows any keypunch symbol. Format Type I allows only integer numbers right-justified in the field.

Note 2: Only nonzero elements need be entered.

As an example of card input to subroutine READIM, consider the following matrix:

$$[A1*C]_{3 \times 6} = \begin{bmatrix} 1 & 0 & 3 & 0 & 6 & 5 \\ 0 & 2 & 4 & 0 & 0 & 0 \\ 0 & 7 & 0 & 0 & 0 & 0 \end{bmatrix}$$

This matrix is also to be written on tape 21 that is to be initialized and identified as T4334. Figure A-3 demonstrates how this information could be written on a coding form to facilitate keypunching to cards.

Tape Data Input Form

In the following, an asterisk denotes required entries. Other entries are optional. Only one card is used for each matrix read.

	<u>Card Columns</u>	<u>Format Type¹</u>	<u>Entry</u>
One Card	1-6	A	*Name of matrix to be read from the Read-Tape
	10		Zero-- Read-Tape will move forward from its present position and search to the end of the tape. If the matrix is not found on the first end-of-tape encounter, the tape will automatically rewind and make one more pass. If it is not found on the second end-of-tape encounter, an error message will be printed and the program will stop.
	7-10	I or	Minus the location number of matrix on the Read-Tape. Tape will be positioned at the beginning of the location specified and will then continue as described above for a zero in column 10.

Figure A-3. Example of card input for subroutine READIM.

	<u>Card Columns</u>	<u>Format Type¹</u>	<u>Entry</u>
One Card	11-15	I	*Read-Tape number (e.g., 11)— If positive, the matrix read will be printed in the output. If negative, the matrix read will not be printed in the output.
	16-21	A	*Run number of matrix to be read from the Read-Tape.
	22-27		REWIND—Read-Tape will be rewound before being used.
	22-25	or	LIST—Read-Tape will be listed by subroutine LTAPE.
	22-27	or	Anything else will be considered as part of the remarks described below.
	28-69	A	Any remarks to further identify the input matrix.
Write-Tape Options	72		\$—Only if the Write-Tape is to be initialized by subroutine INTAPE. The Write-Tape identi- fication will be from card columns 73 through 78.
	72	or	Anything other than \$ if the Write-Tape is not to be initialized.
	73-78	A	Write-Tape identification (e.g., T1234)—Use with \$ in card column 72.
	73-78	or	REWIND—Write-Tape will be rewound before being used.
	73-76	or	LIST—Write-Tape will be listed by subroutine LTAPE after the matrix has been written on the Write-Tape.

<u>Card Columns</u>	<u>Format Type¹</u>	<u>Entry</u>
73-78	or	Anything else will be ignored.
79-80	I	Write-Tape number (e.g., 21)
	or	Blank if the matrix is not to be written on tape.

Note 1: Format Type A allows any keypunch symbol. Format Type I allows only integer numbers right-justified in the field.

As examples of tape input to subroutine READIM, consider:

- Example 1—A matrix named AB2 with run number of RUN-46 is to be read from tape number 11 into the computer and printed. This matrix is also to be written on tape number 22 that is to be initialized and identified as T4321.
- Example 2—A matrix named XYZ4 with run number of TKD is on tape number 13 twice. The first time is at location 29 and the second time is at location 54. It is desired to read the second matrix.

Figure A-4 demonstrates how these two examples would be written on a coding form to facilitate keypunching to cards.

RTAPE

Subroutine RTAPE reads a selected matrix from tape (disk) into the computer core. The matrix to be selected is identified by the desired run number and matrix name. This procedure is accomplished by searching the matrix headings (see subroutine WTAPE writeup) until a match with the desired run number and matrix name is obtained and then by reading the matrix elements from tape (disk) into the computer core. The search begins at the current position (does not rewind) of the tape (disk) and proceeds to the EOT (end of tape defined in subroutine WTAPE writeup). If the desired matrix was not found on reaching the EOT, a rewind is performed, and one more search to the EOT is made. If the desired matrix is again not found: (1) an error message is printed, (2) a listing of the matrix headings is printed (see subroutine LTAPE writeup), and (3) transfer is made to subroutine ZZBOMB where the program is terminated.

START

Subroutine START performs the following operations:

- Reads input card 1 for the run number (any keypunch symbol in card columns 1 through 6) and the user's name (any keypunch symbol in card columns 11 through 28).

If the run number is equal to STOP (card columns 1 through 4), the run is terminated.

If the run number is not equal to STOP, the run continues in subroutine START as follows.

- Reads input card 2 for title card 1. Any keypunch symbols may be used in card columns 1 through 72.
- Reads input card 3 for title card 2. Any keypunch symbols may be used in card columns 1 through 72.
- Initializes page number as zero for use in subroutine PAGEHD.
- Interrogates computer for the date.

Run number, date, page number, user's name, title card 1, and title card 2 are transferred by a COMMON block labeled LSTART for use in other subroutines PAGEHD, PLOT1, PLOT2, PLOT3, and WTape.

Subroutine START is used to start each computer run in the FORMA system and will normally be the first subroutine called in a computer program. For example, pertinent statements from a program using START could be:

```
1.CALL START
      .
      .
      .
      GO TO 1
      END
```

WRITE

Subroutine WRITE writes a matrix of real numbers (a Fortran term for numbers with a decimal point) on paper. A group of up to ten consecutive elements from a row of the matrix are printed on each line. If all elements of a group are zero, printing of this line is suppressed.

Each matrix that is printed begins on a new page. Each page of printout contains the page heading given by subroutine PAGEHD, the name of the matrix, and the row size and column size of the matrix, followed by the matrix data. On any line of matrix data, the first integer number is the row number of the matrix elements on that line. The second integer number is the column number of the matrix element in the first data field. The next group of real numbers (up to ten) are the values of the matrix elements. This group of matrix elements is given in consecutive column order.

WRITIM

Subroutine WRITIM writes a matrix of integer numbers on paper. A group of up to 20 consecutive elements from a row of the matrix are printed on each line. If all elements of a group are zero, printing of this line is suppressed.

Each matrix that is printed begins on a new page. Each page of printout contains the page heading given by subroutine PAGEHD, the name of the matrix, and the row size and column size of the matrix, followed by the matrix data. On any line of matrix data, the first integer number is the row number of the matrix elements on that line. The second integer number is the column number of the matrix element in the first data field. The next group of integer numbers (up to 20) are the values of the matrix elements. This group of matrix elements is given in consecutive column order.

WTAPE

Subroutine WTAPE writes matrix data at the end of existing written matrix data on a FORMA tape. (A disk is preferred; see the following.) Each set of matrix data consists of two logical records. The first record contains the matrix heading (tape identification, location number, run number, matrix name, number of rows of matrix, number of columns of matrix, date, and the word "dense"). The second record consists of the matrix elements.

The following sketch is a schematic representation of the tape (disk):

Beginning
of
tape (disk)



where

H_i = Matrix heading of the i^{th} written matrix

E_i = Matrix elements of the i^{th} written matrix

EOT = End of tape—Data written by subroutine WTAPE or INTAPE that all FORMA tape subroutines recognize as the end of written data.

Each vertical line is an end of logical record put on by the computer system's routines. The tape is written in binary form as opposed to binary coded decimal (BCD) form.

To find the end of written matrix data, a search is made of the matrix headings until the EOT is found. For this reason, a "new" tape (disk) must be initialized with subroutine INTAPE so that the tape (disk) contains an EOT. A "new" tape (disk) is defined as a tape (disk) for which it is desired to start writing matrix data at the front of the tape (disk). Thus, a "new" tape (disk) could be one that contains obsolete FORMA matrix data, as well as one

on which the FORMA system has never written. When the EOT is found, a backspace operation is performed over the EOT, and the current matrix heading, current matrix elements, and a new EOT are then written.

A disk is preferred to a tape for the following reason. Because the physical separation of the read and write heads on most tape drives may cause tolerance problems, backspacing over the EOT is usually not successful. Instead of ending up positioned in front of the EOT, the write head is often positioned in front of the previous matrix elements (E_n in the foregoing sketch). The current matrix heading will be written over the previous matrix elements. This causes problems later when an attempt is made to read the records written on the tape. To alleviate this problem, it is strongly recommended that an intermediate device such as a disk be used with all FORMA tape subroutines (INTAPE, LTAPE, RTAPE, WTAPE, and UPDATE). At the beginning of a computer run, the existing tape should be copied onto the disk by using computer control cards. Likewise, at the end of the run, the disk should be copied back onto tape by using computer control cards.

APPENDIX B

SUPPORT PROGRAMS FOR THE DISCOS FLEXIBLE-BODY CAPABILITY

C					0	1
C					0	2
C				DEBUG # 1	0	3
C					0	4
C	PROGRAM MAIN--DYNAMIC SIMULATION OF CONTROLLED INTERCONNECTED SYSTEM				0	5
C	OF FLEXIBLE BODIES, PROGRAMMED FOR GODDARD SPACE FLIGHT CENTER				0	6
C					0	7
C					0	8
C	CREDITS:				C	9
C	ALL THEORETICAL DEVELOPEMENT AND EQUATION CODEING HAS				C	10
C	BEEN DONE BY:				C	11
C	CARL S. BUDLEY (MARTIN MARIETTA CORPORATION)				0	12
C	A. DARRELL DEVERS (MARTIN MARIETTA CORPORATION)				0	13
C	A. COLTON PARK (MARTIN MARIETTA CORPORATION)				0	14
C					0	15
C	BASIC PROBLEM DEFINITION, REQUIRED STABILITY ANALYSIS				0	16
C	CAPABILITY AND THE BASIC APPROACH TO BE USED FOR DATA				0	17
C	INPUT HAS BEEN PROVIDED BY:				0	18
C	HAROLD P. FRISCH (GODDARD SPACE FLIGHT CENTER)				0	19
C	JAMES H. DONOHUE (GODDARD SPACE FLIGHT CENTER)				0	20
C					C	21
C	THE PROGRAM HAS BEEN REVIEWED AND FURTHER ANNOTATED TO				0	22
C	CROSS REFERENCE WITH THE THEORETICAL MANUAL BY:				0	23
C	HAROLD P. FRISCH (GODDARD SPACE FLIGHT CENTER)				C	24
C					0	25
C	REQUIREMENTS FOR MAKING THE PROGRAM COMPATABLE WITH				0	26
C	THE GSFC IBM-360 SYSTEM WAS DEFINED BY:				0	27
C	REGINALD MITCHELL (GODDARD SPACE FLIGHT CENTER)				0	28
C					0	29
C	THE TECHNICAL MONITOR WAS:				0	30
C	JOSEPH P. YOUNG (GODDARD SPACE FLIGHT CENTER)				C	31
C					0	32
C					0	33
C					0	34
C					0	35
C	NOTE ***** AVOID USE OF QUAD PRECISION AT GSFC				0	36
C	COMPUTATION SPEED AT LEAST 30 TIMES				0	37
C	SLOWER THAN DOUBLE PRECISION				0	38
C					0	39
C	DISCOS SUBROUTINE LOCATION				0	40
C					0	41
C	SUBROUTINE NEW LOCATION OLD LOCATION OVERLAY PRECISION				C	42
C	DEBUG NUMBER TAPE 11702 TAPE 11702				0	43
C					0	44
C	ADDT 105 9331 - 9341 ALPHA-2				0	45
C	ADJB 27 2291 - 2305 BETA -1				0	46
C	ADT 104 9319 - 9330 ALPHA-2				0	47
C	ALPHA A 95 8861 - 8881 ALPHA-4				0	48
C	ASIMLR 85 5819 - 5940 ALPHA-4				0	49
C	ASQR 113 NEW ALPHA-4				0	50
C	EABT 96 8882 - 8918 ALPHA-4				0	51
C	EAKSLV 52 4380 - 4398 ALPHA-2				0	52
C	ECUTGP 51 4236 - 4379 ALPHA-2				0	53
C	EHGENR 45 3766 - 3841 ALPHA-2				0	54
C	ESGENR 47 3875 - 3924 ALPHA-2				0	55
C	ETABA 35 2745 - 2798 BETA -2				0	56
C	CANCDR 78 7368 - 7450 ALPHA-4				0	57
C	CCMENT 7 560 - 590				0	58
C	CONTRL 107 9399 - 9573 ALPHA-2				0	59
C	CREA 21 2021 - 2046 BETA -1				0	60

C	CREADD	19	1859 - 1948	BETA -1	0	61
C	CREB	22	2049 - 2089	BETA -1	0	62
C	CREC	23	2090 - 2218	BETA -1	0	63
C	CREE	20	1949 - 2020	BETA -1	0	64
C	CREMUG	18	1791 - 1858	BETA -1	0	65
C	CRET3	17	1733 - 1790	BETA -1	0	66
C	DCLM2	42	3535 - 3557	ALPHA-2	0	67
C	DCORRT	70	6523 - 6569	ALPHA-4	0	68
C	DEF1	112	NEW (COMMENT CARDS, UNUSED)		0	69
C	DEF2	113	NEW (COMMENT CARDS, UNUSED)		0	70
C	DEF3	114	NEW (COMMENT CARDS, UNUSED)		0	71
C	DEF4	115	NEW (COMMENT CARDS, UNUSED)		0	72
C	DEF5	116	NEW COMMENT CARDS, UNUSED		0	73
C	CFURMB	80	7479 - 7507	ALPHA-4	0	74
C	CIMAG	76	7354 - 7360		0	75
C	DREAL	77	7361 - 7357		0	76
C	DYNS10	2	61 - 308	ALPHA-1	0	77
C	DYNS20	37	2821 - 3005	ALPHA-2	0	78
C	DYNS30	62	5024 - 5096	ALPHA-3	0	79
C	DYNS40	64	5251 - 5818	ALPHA-4	0	80
C	EIGVEC	117	NEW	ALPHA-4	0	81
C	ENGCMC	58	4658 - 4742	ALPHA-2	0	82
C	EQADD	110	9633 - 9669	ALPHA-2	0	83
C	EXTOR	108	9574 - 9603	ALPHA-2	0	84
C	FETCH	25	2239 - 2272	BETA -1	0	85
C	FINDT	66	5541 - 6009	ALPHA-4	0	86
C	FINDU	33	3006 - 3155	ALPHA-2	0	87
C	FIT	75	7332 - 7353	ALPHA-4	0	88
C	FURMB	79	7451 - 7478	ALPHA-4	0	89
C	GAUSSI	31	2484 - 2547		0	90
C	GETBMB	48	3925 - 4022	ALPHA-2	0	91
C	GMISC	111	9661 - 9669	ALPHA-2	0	92
C	GRVGRD	50	4141 - 4235	ALPHA-2	0	93
C	INTAPE	9	619 - 643		0	94
C	INVINP	41	3492 - 3534		0	95
C	KFRINGE	106	9342 - 9398	ALPHA-2	0	96
C	LINEAR	40	3350 - 3491	ALPHA-2	0	97
C	LPLTWR	100	9159 - 9184	ALPHA-4	0	98
C	LPRNT	101	9185 - 9270	ALPHA-4	0	99
C	LTAPE	14	967 - 1029		0	100
C	LTORGL	103	9295 - 9318	ALPHA-4	0	101
C	LTRISP	99	8998 - 9158	ALPHA-4	0	102
C	MAIN	1	1 - 60		0	103
C	NGEN	49	4023 - 4140	ALPHA-2	0	104
C	NRIGID	28	2306 - 2409	ALPHA-1	0	105
C	NSMDDC	15	1030 - 1461	BETA -2	0	106
C	NSMDDL	16	1462 - 1732	BETA -1	0	107
C	MULTA	34	2704 - 2744		0	108
C	MULTAD	55	4431 - 4445		0	109
C	MULTB	97	8919 - 8952	ALPHA-4	0	110
C	MULTSR	56	4446 - 4463	ALPHA-2	0	111
C	MULT3	54	4416 - 4430		0	112
C	NIPLCT	90	8386 - 8516	ALPHA-4	0	113
C	NUMS	68	6325 - 6455	ALPHA-4	0	114
C	NYPLCT	91	8517 - 8629	ALPHA-4	0	115
C	PAGEHD	8	591 - 618		0	116
C	FLUTSS	94	8801 - 8860	ALPHA-4	0	117
C	FLUTWR	61	4977 - 5023	ALPHA-2	0	118
C	FLTCAR	63	5057 - 5250	ALPHA-3	0	119
C	PRNTGU	60	4854 - 4976	ALPHA-2	0	120

C	FRJ	36	2799 - 2020	DELTA -2		0	121
C	GRCON	65	7851 - 7930	ALPHA-4	REAL*16	0	122
C	GRDVR	64	7797 - 7050	ALPHA-4	REAL*16	0	123
C	GRZ	87	8032 - 8113	ALPHA-4	REAL*16	0	124
C	READ	10	644 - 794			0	125
C	READIM	3	369 - 453			0	126
C	REVADD	98	8953 - 8997	ALPHA-4		0	127
C	REVERSE	50	2428 - 2483	DELTA -2		0	128
C	REACAM	59	4743 - 4853	ALPHA-2		0	129
C	FLGCU5	74	7086 - 7331	ALPHA-4		0	130
C	FLPLCT	92	8630 - 8761	ALPHA-4		0	131
C	FMVZRU	71	6570 - 6690	ALPHA-4		0	132
C	FCTDH	44	3583 - 3765	ALPHA-2		0	133
C	FCTDS	40	3842 - 3874	ALPHA-2		0	134
C	FCTFR	32	2584 - 2685			0	135
C	HTAPE	11	795 - 849			0	136
C	HTOP	81	7508 - 7579	ALPHA-4		0	137
C	HWRITE	88	8114 - 8180	ALPHA-4		0	138
C	SATB	20	2273 - 2290	DELTA -1		0	139
C	SCALE	93	8762 - 8800	ALPHA-4		0	140
C	SFREQ2	73	6610 - 7085	ALPHA-4		0	141
C	SHAFTT	109	9604 - 9632	ALPHA-2		0	142
C	SIFT	72	6591 - 6609	ALPHA-4		0	143
C	SKLMVS	53	4399 - 4415			0	144
C	SPLOT	89	8181 - 8385	ALPHA-4		0	145
C	START	6	511 - 559			0	146
C	STORE	24	2219 - 2230	BETA -1		0	147
C	SUBDIA	85	7989 - 8031	ALPHA-4	REAL*16	0	148
C	IFPLY	69	6456 - 6522			0	149
C	IFTYPE	67	6070 - 6324			0	150
C	TORQUE	57	4464 - 4657	ALPHA-2		0	151
C	TRFB	82	7680 - 7743	ALPHA-4		0	152
C	ITFF	83	7744 - 7796	ALPHA-4		0	153
C	UNITY	33	2686 - 2703			0	154
C	WRITE	12	850 - 908			0	155
C	WRITES	43	3558 - 3582			0	156
C	WRITIM	4	434 - 483			0	157
C	WRITIS	5	484 - 510			0	158
C	WTAPE	13	909 - 966			0	159
C	YDUT	59	3156 - 3349	ALPHA-2		0	160
C	YDUTL	102	9271 - 9294	ALPHA-4		0	161
C	ZERC	29	2410 - 2427			0	162
C						0	163
C						0	164
C	IMPLICIT REAL*(A-H,O-Z)					0	165
C						0	166
C						0	167
*	COMMON /ORATIO/					100	168
	IFL1,IFL2,DRVEC(150)					0	169
*	COMMON /GSDATA/					0	170
	GAMG1(3),GMAG,RCMAG					0	171
*	COMMON /ILINER/					0	172
	IFLNER					0	173
*	COMMON /MAXMUM/					0	174
	N3MAX,NHMAX,NSPMAX,NMWMAX,NMWBOD,NMDBOD,CMU,KY,KU					0	175
*	COMMON /MISCNO/					0	176
	NUPRNT,NOPLCT					0	177
*	COMMON /NUMBRS/					0	178
	ZRC,ONE,TWO,TRES					0	179
*	COMMON /PLTDIA/					0	180
	NRFPLOT,NCPLCT					0	180

	COMMON /SPECIF/	0	181
*	BETAH(6, 6),BETAHD(6, 6),AMO(2, 5),RH(3,3,30),RS(3,3,30),	16	182
*	OH(3,35),OS(3,30),IMO(3, 5),NMOW(6, 5),IFISMW(15),	17	183
*	NB,NH,NSPT,NOFMO,NDELTA,ITOPOL(2, 6),IRGFLX(6),IHDATA(7, 6),	18	184
*	LOCU(14),LENU(14),NU,NBETA,NLAM,NEQ	19	185
	COMMON /TAPE/	C	186
*	NTAPE1,NTAPE2,NTAPE3	0	187
C		0	188
	IFL1 = 0	0	189
C		0	190
	NTAPE1 = 1	0	191
	NTAPE2 = 2	0	192
	NTAPE3 = 3	0	193
C	DEFINE MAX SIZES OF ARRAYS. THESE MAY ONLY BE CHANGED VIA	0	194
C	THE REDIMENSION PROGRAM	0	195
	NBMAX = 6	41	196
	NHMAX = 6	42	197
	NSPMAX = 15	43	198
	NMWMAX = 5	44	199
	NMWBCD = 4	45	200
	NMBCD = 12	46	201
	KMU = 22	47	202
	KY = 250	48	203
	KU = 113	49	204
C		0	205
	ZRC = 0.0 0	0	206
	CNE = 1.0 0	0	207
	TWC = 2.0 0	0	208
	TRES = 3.0 0	0	209
C		0	210
999	CALL START	0	211
	CALL COMENT	0	212
C		0	213
	REWIND NTAPE1	0	214
	REWIND NTAPE2	0	215
	REWIND NTAPE3	0	216
C		0	217
C	CALL DYN510 TO READ IN AND PROCESS THE DATA WHICH	0	218
C	DEFINES THE COUPLED BODY SIMULATION MODEL	0	219
	CALL DYN510	0	220
C		0	221
C	CALL DYN520 TO DERIVE AND INTEGRATE THE NON-LINEAR	0	222
C	COUPLED BODY EQUATIONS OF MOTION OR CALL THE ROUTINES	0	223
C	WHICH WILL NUMERICALLY LINEARIZE THE EQUATIONS	0	224
	CALL DYN520	0	225
C		0	226
C	CALL DYN540 TO GO THROUGH ALL LINEAR STABILITY	0	227
C	ANALYSIS OF THE COUPLED EQUATIONS OF MOTION	0	228
	IF (IFLNER .EQ. 1) CALL DYN540	0	229
C		0	230
C	CALL DYN530 ONLY IF PLOTTING IS REQUESTED	0	231
	IF (NOPLGT .GT. 0) CALL DYN530	0	232
C		0	233
	GO TO 999	0	234
C		0	235
	END	0	236

	SUBROUTINE DYN510		0	237
C		DEEUG # 2	0	238
	IMPLICIT REAL*8(A-H,O-Z)		0	239
C			0	240
C			C	241
	COMMON /CONPAR/		0	242
*	CNTCTA(150)	95	243	
	COMMON /GGDATA/	0	244	
*	GAMGI(3),GMAG,RCMAG	0	245	
	COMMON /ILINER/	0	246	
*	IFLNER	0	247	
	COMMON /MAXMUM/	0	248	
*	NBMAX,NHMAX,NSPMAX,NMWMAX,NMWEOB,NMDBJD,<40,KY,KU	0	249	
	COMMON /MISCND/	0	250	
*	NSPENT,NUPLOT	C	251	
	COMMON /NHNS /	0	252	
*	NHPCI(0), NSPUI(0)	12	253	
	COMMON /SPECIF/	0	254	
*	BETAH(6, 6),BETAHD(6, 6),AMO(2, 5),RH(3,3,30),RS(3,3,30),	16	255	
*	OH(3,35),DS(3,30),IMU(3, 5),NMCW(6, 6),IFSMW(15),	17	256	
*	NB,NH,NSPT,NUPMO,NDELTA,ITCPOL(2, 6),IRGFLX(6),IHDATA(7, 6),	18	257	
*	LOCCL(14),LENU(14),NU,NBETA,NLAM,NEQ	19	258	
	COMMON /SUMMRY/	0	259	
*	ASUMRY(15,6),ISUMRY(15,3),KSUMRY	96	260	
	COMMON /TAPENO/	0	261	
*	NTAPE1,NTAPE2,NTAPE3	0	262	
	COMMON /TIMESS/	0	263	
*	STARTI,DELTAI,T,ENDT,IMST	0	264	
C		0	265	
	DIMENSION WV(5),IMDATA(3),IPDATA(3)	0	266	
	EQUIVALENCE (GAMGI(1),WV(1))	0	267	
C		0	268	
C		0	269	
C	NB = NO. OF BODIES	0	270	
C	NH = NO. OF HINGES (LE. NB)	0	271	
C	IRGFLX(L) = (0 IF L IS RIGID), (M(L) IF L IS FLEXIBLE)	0	272	
C	M(L) = NO. OF MODES OF BODY (L)	C	273	
C	L=1,NB	C	274	
C	NMCW(L) = NO. MOMENTUM WHEELS/BODY(L), L=1,NB	0	275	
C	ITCPCL(1,1) = 1 (BODY 1)	0	276	
C	ITCPCL(2,1) = 0 (INERTIAL REFERENCE)	0	277	
C	ITCPCL(1,K) = L1, (K .GT. 1)	0	278	
C	ITCPCL(2,K) = L2, (K .GT. 1)	C	279	
C	BODY(L1) REL. TO BODY(L2), K=1,NH.	0	280	
C	IHCATA(1,K) = ITYPE (EULER PERMUTATION 1,2,...12), K=1,NH	0	281	
C	(J,K) = 0 (FORCED/FREE)	0	282	
C	(J,K) = 1 (FIXED CONSTRAINT)	0	283	
C	(J,K) = 2 (RHEGNOMIC CONSTRAINT), -- J=2,7, K=1,NH	C	284	
C	BETAH(J,K) = INITIAL HINGE COORDINATES, (J=1,6, K=1,NH)	0	285	
C	BETAHD(J,K) = INITIAL HINGE RATES, (J=1,6, K=1,NH)	0	286	
C	AM(L) = MASS OF BODY(L), L=1,NB	0	287	
C	SMCM(1,L) = STATIC MOMENT OF BODY(L), L=1,NB	0	288	
C	AIN(1,L) = MOMENT OF INERTIA PROPERTIES OF BODY(L), I=1,6, L=1,NB	C	289	
C		C	290	
C		C	291	
	1001 FORMAT (10I5)	0	292	
	2001 FORMAT (//15X48HSUMMARY OF DYNAMIC-SIMULATION-PROGRAM INPUT DATA,	0	293	
*	10(2H *),//3X12HACTUAL SIZES5X13HMAXIMUM SIZES4X12HINTEGRATION	0	294	
*	4HDATA 12X21HGRAVITY GRADIENT DATA17X10HMISC. DATA,	0	295	
*	/3X13(1H-),4X13(1H-),4X16(1H-),5X37(1H-),6X11(1H-),	0	296	

```

* /3X9HNE      = 14,4X9HNBMAX = 14,4X9HSTARTT = 1PD10.3,4X7HG1      = 0 297
* 1PD10.3,4X8HGAMA1 = 1PD10.3,4X9HNOPRNT = 12,                      0 298
* /3X9HNNH      = 14,4X9HNNHMAX = 14,4X9HDELTA = 1PD10.3,4X7HG2      = 0 299
* 1PD10.3,4X8HGAMA2 = 1PD10.3,4X9HNCPLT = 12,                      0 300
* /3X9HNSPT      = 14,4X9HNSPMAX = 14,4X9HENUF = 1PD10.3,4X7HG3      = 0 301
* 1PD10.3,4X8HGAMA3 = 1PD10.3,4X9HIFLNER = 12,                      0 302
* /3X9HNCFMO      = 14,4X9HNNMMAX = 14,                          27X7HGMAG = 0 303
* 1PD10.3,4X8HRCMAG = 1PD10.3,                      0 304
* /3X9HNDDELTA = 14,4X9HNNMBUD = 14,                      0 305
* /3X9HNL      = 14,4X9HNNMBUD = 14,                      0 306
* /3X9HNBETA = 14,4X9HKMU = 14,                      0 307
* /3X9HNLAM = 14,4X9HKY = 14,                      0 308
* /3X9HNEQ = 14,4X9HKU = 14,                      0 309
2002 FORMAT (//1X49HTHE TOPLOGY ARRAY (ITOPOL) FOR THIS CASE FOLLOWS) 0 310
2003 FORMAT (//1X50HTHE CONSTRAINT SPECIFICATIONS FOR THIS CASE FOLLOW) 0 311
2004 FORMAT (//1X39HTHE SPECIFIED INITIAL HINGE ANGLES AND              0 312
* 28HDISPLACEMENTS (BETAH) FOLLOW) 0 313
2005 FORMAT (//1X49HTHE SPECIFIED INITIAL HINGE RATES (BETAHD) FOLLOW) 0 314
2006 FORMAT (//1X45HTHE NO. OF ELASTIC MODES/BOJY ARRAY (IRGLX)      0 315
* 7HFCLLWS) 0 316
2007 FORMAT (//1X47HTHE NO. OF P/Q HINGE POINTS/BOJY ARRAY (NHPOI) 0 317
* 7HFCLLWS) 0 318
2008 FORMAT (//1X44HTHE NO. OF SENSOR PCINTS/BOJY ARRAY (NSPCI)      0 319
* 7HFCLLWS) 0 320
2009 FORMAT (//1X40HTHE MUM. WHEEL/BCDY TABLE (NMGW) FOLLOWS )      0 321
2010 FORMAT (//1X44HTHE STATE VECTOR LENGTH ARRAY (LENU) FOLLOWS )    0 322
2011 FORMAT (//1X46HTHE STATE VECTOR LOCATION ARRAY (LUCU) FOLLOWS)    0 323
2012 FORMAT (//1X50HTHE SPECIFIED SENSOR POINT/BOJY CORRELATION ARRAY 0 324
* 16H(IFTSMW) FOLLOWS ) 0 325
2013 FORMAT (//1X43HTHE FOLLOWING DATA IS SPECIFIED MUM. WHEEL        0 326
* 47HINFORMATION (IF ANY) AND CONTROLLER INFORMATION /1X90(1H-)) 0 327
2014 FORMAT (//1X45HTHE SPECIFIED MCM. WHEEL CONTROL ARRAY (IMO)      0 328
* 7HFCLLWS) 0 329
2015 FORMAT (//1X50HTHE SPECIFIED MCM. WHEEL RATES AND INERTIAS (AMO) 0 330
* 7HFCLLWS ) 0 331
2016 FORMAT (//1X43HTHE SPECIFIED CONTROLLER INITIAL CONDITIONS AND    0 332
* 22HCHARACTERISTICS FOLLOW /3X30H(THE FIRST 4DELTA ARE INITIAL      0 333
* 38HCONTROLLER STATE VARIABLES, THERE ARE 13,12H ADDITIONAL        0 334
* 19HCONTROL PARAMETERS)) 0 335
2110 FORMAT (1H1, 28HINPUT DATA ERROR, NERROR = 13)                  0 336
C 0 337
DATA NIT,NUT / 5, 6/ 0 338
C 0 339
CCCCC 0 340
KCONT = 150 96 341
KSLMFY = 15 99 342
CCCCC 0 343
C 0 344
C***** READ BASIC SYSTEM SIZES 0 345
READ (NIT,1001) NB,NH,NSPT,NOFMC,NDELTA 0 346
C 0 347
NERROR = 1 0 348
C***** READ SYSTEM TOPLOGY 0 349
CALL READIM (ITOPOL,N1,N2,2,NHMAX) 0 350
IF (N1.NE.2 .OR. N2.NE.NH .OR. NH.LT.NB) GO TO 999 0 351
NERROR = 2 0 352
C***** READ BOJY TYPE AND NUMBER OF MJDES IF FLEXIBLE 0 353
CALL READIM (IRGLX,N1,N2,1,NBMAX) 0 354
IF (N2 .NE. NB) GO TO 999 0 355
NERROR = 3 0 356

```

```

C***** READ SENSOR POINT LOCATION INFORMATION                                0 357
CALL READIM (IFTSMA,N1,N2,1,NSPMAX)                                         0 358
IF (N2.NE.NSPT) GO TO 999                                                    C 359
                                                                           NERROR = 4 0 360
C***** READ CONSTRAINT DATA FOR EACH HINGE POINT                          0 361
CALL READIM (IHDATA,N1,N2,7,NHMAX)                                          0 362
IF (N1.NE.7 .OR. N2.NE.NH) GO TO 999                                        0 363
                                                                           NERROR = 5 0 364
C***** READ CONTIGUOUS HINGE FRAME ORIENTATION DATA                      0 365
CALL READ (BETAH,N1,N2,6,NHMAX)                                             0 366
IF (N1.NE.6 .OR. N2.NE.NH) GO TO 999                                        0 367
                                                                           NERROR = 6 0 368
C***** READ CONTIGUOUS HINGE FRAME RATE DATA                            0 369
CALL READ (BETAH,N1,N2,6,NHMAX)                                             0 370
IF (N1.NE.6 .OR. N2.NE.NH) GO TO 999                                        0 371
                                                                           NERROR = 7 0 372
IF (ITOPCL(1,1).NE.1 .OR. ITOPCL(2,1).NE.0) GO TO 999                    0 373
                                                                           NERROR = 8 0 374
DO 605 J=2,NH                                                                0 375
IF (ITOPCL(1,J) .EQ. ITOPCL(2,J)) GO TO 999                                0 376
605 CONTINUE                                                                  0 377
C                                                                              0 378
CCC PRELIMINARY TOPOLOGY CHECK. COMPLETE CHECK DONE BY ROTDH ....          0 379
                                                                           NERROR = 9 0 380
DO 610 N=1,NB                                                                0 381
DO 615 I=1,2                                                                0 382
DO 615 J=2,NH                                                                0 383
IF (ITOPCL(1,J) .EQ. N) GO TO 610                                          0 384
615 CONTINUE                                                                  0 385
GO TO 999                                                                    0 386
610 CONTINUE                                                                  0 387
                                                                           NERROR = 10 0 388
DO 620 J=1,NB                                                                0 389
IF (IRGFLX(J).LT.0 .OR. IRGFLX(J).GT.NMCBJJ) GO TO 999                  0 390
620 CONTINUE                                                                  0 391
                                                                           NERROR = 11 0 392
DO 625 J=1,NSPT                                                             0 393
IF (IFTSMA(J).LT.1 .OR. IFTSMW(J).GT.NB) GO TO 999                      0 394
625 CONTINUE                                                                  0 395
                                                                           NERROR = 12 0 396
DO 630 J=1,NH                                                                0 397
IF (IHDATA(1,J).LT.1 .OR. IHDATA(1,J).GT.12) GO TO 999                  0 398
DO 630 I=2,7                                                                0 399
IF (IHDATA(1,J).LT.0 .OR. IHDATA(1,J).GT.2) GO TO 999                    0 400
630 CONTINUE                                                                  0 401
C                                                                              0 402
C                                                                              0 403
C                                                                              0 404
C                                                                              0 405
C                                                                              0 406
C                                                                              0 407
C                                                                              0 408
C                                                                              0 409
C                                                                              0 410
C                                                                              0 411
C                                                                              0 412
C                                                                              0 413
C                                                                              0 414
C                                                                              0 415
C                                                                              0 416
DO 5 N=1,NB                                                                0 406
NHPOI(N) = 0                                                                0 407
NSPOI(N) = 0                                                                C 408
DO 10 I=1,2                                                                0 409
DO 10 J=2,NH                                                                0 410
IF (ITOPCL(1,J) .EQ. N) NHPOI(N) = NHPOI(N) + 1                        0 411
10 CONTINUE                                                                  0 412
DO 15 J=1,NSPT                                                             0 413
IF (IFTSMA(J) .EQ. N) NSPOI(N) = NSPOI(N) + 1                          0 414
15 CONTINUE                                                                  0 415
5 CONTINUE                                                                  0 416

```


C		C	417	
	11 = NMWBDQ + 2	C	418	
	DO 4C I=1,11	C	419	
	DO 4C J=1,NBMAX	C	420	
40	NMCW(I,J) = 0	C	421	
	IF (NCFMG .EQ. 0) GO TO 41	C	422	
		NERROR = 13	C	423
C*****	READ BASIC MOMENTUM WHEEL DATA	C	424	
	CALL READIM (IMO,N1,N2, J,NMWMAX)	C	425	
	IF (N1.NE.3 .OR. N2.NE.NCFMU) GO TO 999	C	426	
		NERROR = 14	C	427
C*****	READ MOMENTUM WHEEL RATE + INERTIA	C	428	
	CALL READ (AMO,N1,N2, 2,NMWMAX)	C	429	
	IF (N1.NE.2 .OR. N2.NE.NCFMU) GO TO 999	C	430	
		NERROR = 15	C	431
	DO 635 J=1,NCFMU	C	432	
	IF (IMU(1,J).LT.1 .OR. IMU(1,J).GT.NSPT) GO TO 999	C	433	
	IF (IMU(2,J).LT.1 .OR. IMU(2,J).GT.3) GO TO 999	C	434	
	IF (IMU(3,J).LT.0 .OR. IMU(3,J).GT.1) GO TO 999	C	435	
	IF (AMO(2,J) .LE. 0.D 0) GO TO 999	C	436	
635	CONTINUE	C	437	
		NERROR = 16	C	438
	IF (NCFMG .EQ. 1) GO TO 641	C	439	
	N1 = NOFMD - 1	C	440	
	DO 640 I=1,N1	C	441	
	IP1 = I + 1	C	442	
	DO 640 J=IP1,NOFMD	C	443	
	IF (IMO(1,I) .EQ. IMO(1,J)) GO TO 999	C	444	
640	CONTINUE	C	445	
		NERROR = 17	C	446
C			C	447
	NMCW = MOMENTUM WHEEL/BODY TABLE	C	448	
C		C	449	
641	DO 45 N=1,NCFMU	C	450	
	NPTS = IMO(1,N)	C	451	
	NBCD = IFTSM*(NPTS)	C	452	
	NMCW(1,NBDQ) = NMOW(1,NBDQ) + 1	C	453	
	IF (IMU(3,N) .NE. 0) NMCW(2,NBDQ) = NMOW(2,NBDQ) + 1	C	454	
	IC = NMCW(1,NBDQ) + 2	C	455	
	IF (IC .GT. 11) GO TO 999	C	456	
45	NMCW(IC,NBDQ) = N	C	457	
41	CONTINUE	C	458	
C		C	459	
	LENU = STATE VECTOR LENGTH ARRAY	C	460	
C		C	461	
	LOCU = STATE VECTOR LOCATION ARRAY	C	462	
C		C	463	
	NEC = 0	C	464	
	DO 50 N=1,NB	C	465	
	NEC = NEC + IRGFLX(N)	C	466	
50	LENU(N) = 6 + IRGFLX(N) + NMOW(2,N)	C	467	
	LOCU(1) = 1	C	468	
	DO 55 N=2,NB	C	469	
55	LOCU(N) = LOCU(N - 1) + LENU(N - 1)	C	470	
	NU = LOCU(NB) + LENU(NB) - 1	C	471	
	DO 56 N=1,NB	C	472	
56	LENU(N+NB) = IRGFLX(N)	C	473	
	DO 57 N=1,NB	C	474	
	NM1 = N - 1	C	475	
57	LOCU(N+NB) = LOCU(NM1+NB) + LENU(NM1+NB)	C	476	
	NBETA = 0			
	NLAM = 0			

DO 6C I=2,7	0	477
DO 6C J=1,NH	0	478
IF (IHDATA(1,J) .NE. 1) NBETA = NBETA + 1	0	479
IF (IHDATA(1,J) .NE. 0) NLAM = NLAM + 1	0	480
6C CONTINUE	0	481
NEQ = NEG + NBETA + NDELTA + NU	0	482
LENU(2*NB+1) = NBETA	0	483
LENU(2*NB+2) = NDELTA	0	484
LOCU(2*NB+1) = LOCU(2*NB) + LENU(2*NB)	0	485
LOCU(2*NB+2) = LOCU(2*NB+1) + NBETA	0	486
C	0	487
C***** READ TIME HISTORY INTEGRATION CONTROL	0	488
CALL READ (TMDATA,N1,N2,1,3)	0	489
C	0	490
C***** READ PRINT, PLOT, LINEAR, NON-LINEAR FLAGS	0	491
CALL READIM (IPDATA,N1,N2,1,3)	0	492
C	0	493
C***** READ CONTROL SYSTEM DATA	0	494
CALL READ (CNTDATA,N1,NCONPAK,1,KCONT)	0	495
	NERROR = 18	0
C***** READ GRAVITY GRADIENT DATA	0	497
CALL READ (WV,N1,N2,1,5)	0	498
C	0	499
IF (N2 .NE. 4) GO TO 999	0	500
RCMAG = WV(4)	0	501
GMAG = DSQRT(GAMGI(1)**2 + GAMGI(2)**2 + GAMGI(3)**2)	0	502
CC WV(5) IS NOW RCMAG, WV(4) IS GMAG	0	503
IF (GMAG .EQ. 0.D 0) GO TO 75	C	504
IF (RCMAG .LE. 1.D 0) GO TO 999	0	505
C COMPUTE UNIT VECTOR, INERTIAL COORDINATES IN DIRECTION	C	506
C OF GRAVITY FIELD	0	507
DO 7C J=1,3	0	508
7C GAMGI(J) = GAMGI(J)/GMAG	0	509
C	0	510
75 CALL PAGEHD	0	511
STARTT = TMDATA(1)	0	512
LELO = 2*NB + 2	0	513
DELTAT = TMDATA(2)	0	514
ENCT = TMDATA(3)	0	515
NCPRT = IPDATA(1)	C	516
NOPLCT = IPDATA(2)	0	517
IFLNER = IPDATA(3)	0	518
G1 = GMAG*GAMGI(1)	0	519
G2 = GMAG*GAMGI(2)	0	520
G3 = GMAG*GAMGI(3)	0	521
WRITE (NCT,2001) NB,NBMAX,STARTT,G1,GAMGI(1),NCPRT,NH,NHMAX,	0	522
* DELTAT,G2,GAMGI(2),NOPLCT,NSPT,NSPMAX,ENCT,3,3,GAMGI(3),IFLNER,	0	523
* NDFMD,NMWMAX,GMAG,RCMAG,NDELTA,NMWBOD,NO,NMWBOD,NBETA,KMU,	0	524
* NLAM,KY,NEQ,KU	0	525
WRITE (NCT,2002)	0	526
CALL WRITIS (IFOPUL,2,NH,2)	0	527
WRITE (NCT,2003)	0	528
CALL WRITIS (IHDATA,7,NH,7)	0	529
WRITE (NCT,2004)	C	530

CALL WRITES (BETAH,6,NH,6)	0	531
WRITE (NCT,2005)	0	532
CALL WRITES (BETAHD,6,NH,6)	0	533
CALL PAGEHD	0	534
WRITE (NCT,2006)	0	535
CALL WRITIS (IRGFLX,1,NB,1)	0	536
WRITE (NCT,2007)	0	537
CALL WRITIS (NHPOI,1,NB,1)	C	538
WRITE (NCT,2008)	0	539
CALL WRITIS (NSPOI,1,NB,1)	C	540
WRITE (NCT,2009)	0	541
CALL WRITIS (NMDW,11,NB,11)	0	542
WRITE (NCT,2010)	C	543
CALL WRITIS (LENU,1,LELO,1)	0	544
WRITE (NCT,2011)	C	545
CALL WRITIS (LOCU,1,LELO,1)	C	546
WRITE (NCT,2012)	0	547
CALL WRITIS (IFTSMW,1,NSPT,1)	0	548
CALL PAGEHD	C	549
WRITE (NCT,2013)	0	550
IF (NOFMC .NE. 0) WRITE (NCT,2014)	0	551
IF (NOFMC .NE. 0) CALL WRITIS (IMO,3,NOFMC,3)	0	552
IF (NOFMC .NE. 0) WRITE (NCT,2015)	0	553
IF (NOFMC .NE. 0) CALL WRITES (AMO,2,NOFMC,2)	0	554
NCNTRL = NCNPAR - NOELTA	0	555
WRITE (NCT,2016) NCNTRL	0	556
CALL WRITES (CNTDTA,1,NCNPAR,1)	C	557
C	0	558
C GET EDDY DATA FOR EACH OF THE NB BODIES	0	559
DO 20 N=1,NB	0	560
IF (IRGFLX(N) .EQ. 0) GO TO 25	0	561
C	0	562
C***** READ FLAG TO SPECIFY LUMPED OR CONSISTANT MASS	0	563
C***** FLEXIBLE BODY DATA	0	564
READ (NIT,1001) NTYPE	0	565
C	0	566
C MSMODL PROCESSES LUMPED MASS DATA FOR FLEXIBLE BODIES	0	567
IF (NTYPE .EQ. 1) CALL MSMODL (N)	0	568
C	0	569
C MSMODC PROCESSES CONSISTANT MASS DATA FOR FLEXIBLE BODIES	0	570
IF (NTYPE .EQ. 2) CALL MSMODC (N)	C	571
C	0	572
GO TO 20	0	573
C MRIGID PROCESSES DATA FOR RIGID BODIES	C	574
25 CALL MRIGID (N)	0	575
C	0	576
20 CONTINUE	0	577
C	0	578
RETURN	0	579
999 WRITE (NCT,2110) NEKRUR	C	580
STGP	0	581
END	C	582

SUEROUTINE READIM (IA,KR,NC,KR,KC)	C	583
C	DEBUG # 3	0 584
REAL*8 ANAME,REMRK		0 585
DIMENSION IA(KR,1),IX(14),KLMRK(9)		0 586
DATA NIT,NOT,IASTRS/ 5, 6, 1H* /		0 587
C		0 588
CC--TAKEN FROM FORMA.		0 589
CC--ACKNOWLEDGEMENT GIVEN TO RF HRUDA.		0 590
C		0 591
1001 FORMAT (A6,I4,I5,9A6, 2XAI)		0 592
1002 FORMAT (16I5)		0 593
2001 FORMAT (/27H CARD INPUT INTEGER MATRIX A6, 2X 1H(I4,2H X I4,2H)		0 594
* 2X 9A6,//)		0 595
2002 FORMAT (/27H CARD INPUT INTEGER MATRIX A6, 2X 1H(I4,2H X I4,2H)		0 596
* 3X 9HCONTINUED //)		0 597
2004 FORMAT (1X 16I5)		0 598
2005 FORMAT (15H0END OF READIM.)		0 599
2006 FORMAT (1H1,37HERROR IN SUBROUTINE READIM, NERROR = ,I3)		0 600
C		0 601
C READ IN HEADER CARD.		0 602
READ (NIT,1001) ANAME,N1,N2,REMRK,IZI		0 603
IPRIN = 0		0 604
IF (IZI .EQ. IASTRS) IPRIN = 1		0 605
IF (IPRIN .EQ. 1) CALL PAGEHD		0 606
C		0 607
C CARD READING SECTION.		0 608
NR = N1		0 609
NC = N2		0 610
IF (IPRIN .EQ. 1) WRITE (NOT,2001) ANAME,NR,NC,REMRK		0 611
NERROR = 1		0 612
IF (NR.GT.KR .OR. NC.GT.KC) GO TO 999		0 613
NLINE = 0		0 614
DO 105 I=1,NR		0 615
DO 105 J=1,NC		0 616
105 IA(I,J) = 0		0 617
110 READ (NIT,1002) I,JS,IX		0 618
IF (I.EQ.0 .AND. JS.EQ.0) GO TO 400		0 619
NERROR = 2		0 620
IF (I.LE.0 .OR. I.GT.NR .OR. JS.LE.0 .OR. JS.GT.NC) GO TO 998		0 621
JE = JS + 13		0 622
IF (JE .LE. NC) GO TO 115		0 623
JX = NC - JS + 2		0 624
NERROR = 3		0 625
DO 112 J=JX,14		0 626
IF (IX(J) .NE. 0) GO TO 996		0 627
112 CONTINUE		0 628
JE = NC		0 629
115 N = 0		0 630
DO 120 J=JS,JE		0 631
N = N + 1		0 632
120 IA(I,J) = IX(N)		0 633
NLINE = NLINE + 1		0 634
IF (NLINE .LE. 47) GO TO 125		0 635
IF (IPRIN .EQ. 1) CALL PAGEHD		0 636
IF (IPRIN .EQ. 1) WRITE (NOT,2002) ANAME,NR,NL		0 637
NLINE = 1		0 638
125 IF (IPRIN .EQ. 1) WRITE (NOT,2004) I,JS,(IA(I,J),J=JS,JE)		0 639
GO TO 110		0 640
C		0 641
400 IF (IPRIN .EQ. 1) WRITE (NOT,2005)		0 642
RETURN		0 643
C		0 644

998 WRITE (NCT,2004) 1,JS,IX	0	645
999 WRITE (NCT,2006) NERROR	0	646
STOP	0	647
END	0	648
SUBROUTINE WRITIM (IA,NR,NC,ANAME,KR)	0	649
C	DEBUG # 4	0 650
REAL*8 ANAME		0 651
DIMENSION IA(KR,1)		0 652
DATA NCT / 6 /		0 653
C		0 654
CC-- TAKEN FROM FORMA --		0 655
CC -- ACKNOWLEDGEMENT GIVEN TO R L WOHLER.		0 656
C		0 657
2010 FORMAT (//15H OUTPUT MATRIX A6,2X 1H(14,2H X 14,2H) //		0 658
* 16X,20(1X,1H(12,1H))//)		0 659
2020 FORMAT (//15H OUTPUT MATRIX A6,2X 1H(14,2H X 14,2H)		0 660
* 3X, 9HCONTINUED //16X,20(1X,1H(12,1H))//)		0 661
2030 FORMAT (1X,215,5X,2015)		0 662
2040 FORMAT (15HEND OF WRITIM.)		0 663
C		0 664
C PULL UP A NEW PAGE FOR MATRIX AND PRINT MATRIX NAME.		0 665
CALL PAGEHD		0 666
WRITE (NCT,2010) ANAME,NR,NC,(L,L=1,20)		0 667
NLINE = 0		0 668
C		0 669
DO 60 I=1,NR		0 670
NZERC = 0		0 671
JS = 1		0 672
10 JE = JS + 19		0 673
IF (JE .GT. NC) JE = NC		0 674
C SEE IF ELEMENTS ARE ZERO.		0 675
DO 20 J=JS,JE		0 676
IF (IA(I,J) .NE. 0) GO TO 30		0 677
20 CONTINUE		0 678
GO TO 40		0 679
30 NLINE = NLINE + 1		0 680
IF (NLINE .LE. 44) GO TO 35		0 681
CALL PAGEHD		0 682
WRITE (NCT,2020) ANAME,NR,NC,(L,L=1,20)		0 683
NLINE = 1		0 684
35 WRITE (NCT,2030) 1,JS,(IA(I,J), J=JS,JE)		0 685
NZERC = 1		0 686
40 IF (JE .EQ. NC) GO TO 50		0 687
JS = JS + 20		0 688
GO TO 10		0 689
C SKIP A SPACE BETWEEN EACH ROW IF THERE ARE MORE THAN 20 COLUMNS		0 690
C AND SOMETHING HAS BEEN WRITTEN.		0 691
50 IF (NC.LE.20 .OR. NZERC.EQ. 0 .CR. 1.EQ.NR) GO TO 60		0 692
NLINE = NLINE + 1		0 693
WRITE (NCT,2030)		0 694
60 CONTINUE		0 695
C		0 696
WRITE (NCT,2040)		0 697
RETURN		0 698
END		0 699

C	SUBROUTINE WRITIS (IM,NR,NC,KR)		0	700
C		DEBUG # 5	0	701
C	PRINT INTEGER TWO DIMENSIONAL ARRAYS IN FORMA FORMAT		0	702
	DIMENSION IM(KR,I), ICH(20)		0	703
	DATA NOT / 6 /		0	704
	DATA ICH /		0	705
	* 4H(1),4H(2),4H(3),4H(4),4H(5),4H(6),4H(7),4H(8),		0	706
	* 4H(9),4H(10),		0	707
	* 4H(11),4H(12),4H(13),4H(14),4H(15),4H(16),4H(17),4H(18),		0	708
	* 4H(19),4H(20) /		0	709
C			0	710
	2001 FORMAT (17X20(1X,A4))		0	711
	2002 FORMAT (1X,2I5,5X,20I5)		0	712
C			0	713
	LR = 20		0	714
	IF (NC .LT. LR) LR = NC		0	715
	WRITE (NCT,2001) (ICH(L),L=1,LR)		0	716
	DO 60 I=1,NR		0	717
	JS = 1		0	718
10	JE = JS + 19		0	719
	IF (JE .GT. NC) JE = NC		0	720
	WRITE (NCT,2002) I,JS, (IM(I,J),J=JS,JE)		0	721
	IF (JE .EQ. NC) GO TO 60		0	722
	JS = JS + 20		0	723
	GO TO 10		0	724
60	CONTINUE		0	725
C			0	726
	RETURN		0	727
	END		0	728

	SUBROUTINE START		0	729
C		DEBUG # 6	0	730
	REAL*8 IRUNNO,IDATE,UNAME,TITLE1,TITLE2,STOP		0	731
	COMMON /LSTART/IRUNNO,IDATE,NPAGE		0	732
	COMMON /LSTRT1/ UNAME(5),TITLE1(12),TITLE2(12)		0	733
	COMMON /LZMODE/ AMODE(200)		0	734
	DATA NIT,NOT,STOP / 5, 6, 4HSTOP/		0	735
	DATA IIST/ 0 /		0	736
	LOGICAL DEBUG		0	737
	REAL*8 RDEBUG		0	738
	COMMON /LDEBUG/ DEBUG(120)		0	739
	DATA RDEBUG/6HDEBUG /		0	740
C			0	741
CCC	--- TAKEN FROM FORMA. ACKNOWLEDGEMENT GIVEN TO R. HRODA, R. WOHLER		0	742
C	READS INPUT CARD 1 FOR IRUNNO, UNAME.		0	743
C	CHECKS IRUNNO FOR STOP (I.E. IF IRUNNO = STOP, PROGRAM WILL		0	744
C	BE STOPPED). YOU SHOULD HAVE A STOP CARD (THE WORD STOP PUNCHED		0	745
C	STARTING IN COLUMN 1) AFTER YOUR REGULAR DATA DECK.		0	746

C	IF IRUNNO IS NOT EQUAL TO STOP, THE SUBROUTINE CONTINUES AS FOLLOWS.	0	747
C	READS INPUT CARD 2 FOR TITLE1.	0	748
C	READS INPUT CARD 3 FOR TITLE2.	0	749
C	SETS NPAGE = 0.	0	750
C	INTERGATES COMPUTER TO DEFINE DATE AS AN A6.	0	751
C		0	752
C	INPUT ORDER	0	753
C	IRUNNC,UNAME FORMAT (A6, 4X 3A6)	0	754
C	TITLE1 FORMAT (12A6)	0	755
C	TITLE2 FORMAT (12A6)	0	756
C		0	757
C	DEFINITIONS	0	758
C	IRUNNC = RUN NUMBER. (A6 FORMAT)	0	759
C	DATE = DATE. (A8 FORMAT).	0	760
C	NPAGE = PAGE NUMBER.	0	761
C	UNAME = USERS NAME. (3A6 FORMAT)	0	762
C	TITLE1 = FIRST TITLE. (12A6 FORMAT)	0	763
C	TITLE2 = SECOND TITLE. (12A6 FORMAT)	0	764
C		0	765
C	IF IRUNNO = 'DEBUG' THE DEBUG PRINTS CALLED	0	766
C	DEBUG(K) = .FALSE. SKIP DEBUG PRINT IN SUBROUTINE K	0	767
C	.TRUE. PRINT	0	768
C		0	769
	1001 FORMAT (A6, 4X 3A6)	0	770
	1002 FORMAT (12A6)	0	771
	1003 FORMAT (60L1)	0	772
	2003 FORMAT (30H1END OF INPUT DATA HAS BEEN REACHED.)	0	773
C		0	774
	IF (IIST.EQ. 0) CALL MODESG (AMODE,0)	0	775
	IIST = 1	0	776
	READ (NIT,1001) IRUNNO,UNAME	0	777
	DO 12 I=1,120	0	778
12	DEBUG(I) = .FALSE.	0	779
	IF (IRUNNC.NE.ROBUG) GO TO 11	0	780
	READ (NIT,1003) (DEBUG(I),I=1,120)	0	781
11	CONTINUE	0	782
	IF (IRUNNO.NE. STOP) GO TO 10	0	783
	CALL EXITG (AMODE)	0	784
	WRITE (NIT,2003)	0	785
	STOP	0	786
C		0	787
10	READ (NIT,1002) TITLE1	0	788
	READ (NIT,1002) TITLE2	0	789
	NPAGE = 0	0	790
	CALL DATE (1DATE)	0	791
	RETURN	0	792
	END	0	793
	SUBROUTINE COMMENT	0	794
C		0	795
	REAL*8 IREMRK,IPGHD,P,QUIT	0	796
	DIMENSION IREMRK(13)	0	797
	DATA NIT,N01,QUIT,P / 5, 6, 6H000000, 1HP/	0	798
C		0	799
C	READ COMMENT CARDS AND PRINT THEM UNDER PAGE HEADING OF FORMA	0	800

C	SUBROUTINE PAGEHD. COMMENT CARDS MAY HAVE ANY KLYPUNCH SYMBOL	0	801
C	IN CARD COLUMNS 1-78.	C	802
C	IF IT IS DESIRED TO HAVE ANY GIVEN COMMENT CARD PRINT ON A NEW	C	803
C	PAGE, SUPPLY THE LETTER P IN COLUMN 80 ON THAT CARD.	0	804
C	ROUTINE IS ENDED BY SUPPLYING A CARD WITH ZERJS IN COLUMNS 1 THRU 10.	0	805
C	CALLS PERMA SUBROUTINE PAGEHD.	0	806
C	CODED BY RF HRUDA. MARCH 1966.	0	807
C	LAST MODIFICATION BY J ERNST. JUNE 1971.	0	808
C		0	809
	1001 FORMAT (13A6,1X,A1)	0	810
	2001 FORMAT (////)	0	811
	2002 FORMAT (22X,13A6)	0	812
C		0	813
	N = 0	C	814
1	READ (NIT,1001) (IREMRK(I),I=1,13),IPGHD	0	815
	IF (IREMRK(1).EQ.QUIT) RETURN	0	816
	N = N+1	0	817
	IF (N.NE.1 .AND. IPGHD.NL.P) GO TO 2	0	818
	CALL PAGEHD	C	819
	WRITE (NCT,2001)	0	820
	N = 1	0	821
2	IF (N.EC.50) N = 0	C	822
	WRITE (NCT,2002) (IREMRK(I),I=1,13)	0	823
	GO TO 1	0	824
	END	0	825
	SUBROUTINE PAGEHD	0	826
C		0	827
	REAL*8 IRUNNO,DATE,UNAME,TITLE1,TITLE2,TIM	0	828
	COMMON /LSTART/IRUNNO,DATE,NPAGE	0	829
	COMMON /LSTRT1/ UNAME(3),TITLE1(12),TITLE2(12)	0	830
	DATA NCT / 6 /	0	831
C		0	832
C	ERINGS UP NEW PAGE AND PUTS HEADING AT TOP.	0	833
C	BY COMBINED EFFORT OF BODLEY, HRUDA, WOHLER. DEC 1968.	0	834
C	LAST REVISION BY JW ERNST. SEP 1972.	0	835
C		0	836
C	INTERNAL VARIABLES. (TRANSFERRED THRU COMMON).	C	837
C	IRUNNO = RUN NUMBER. (A6 FORMAT)	0	838
C	DATE = DATE. (A8 FORMAT)	0	839
C	NPAGE = PAGE NUMBER.	0	840
C	UNAME = USER'S NAME. (3A6 FORMAT)	0	841
C	TITLE1 = FIRST TITLE. (12A6 FORMAT)	0	842
C	TITLE2 = SECOND TITLE. (12A6 FORMAT)	0	843
C		0	844
	2001 FORMAT (SHIRUN NO. A6, 42X SHDATE A8, 42X 9HPAGE NO. I4,	0	845
	* / 55X 7HRUN BY 3A6, // 10X 12A6,10X15HCURRENT TIME = ,A8,	0	846
	* / 10X 12A6,10X16HTHE CPU TIMER = ,1PE10.4)	0	847
C		0	848
	NPAGE = NPAGE + 1	0	849
	CALL TIME (TIM)	0	850
	CALL SECOND (SEC)	0	851
	WRITE (NCT,2001) IRUNNO,DATE,NPAGE,UNAME,TITLE1,TIM,TITLE2,SEC	0	852
	RETURN	0	853
	END	0	854

C	SUBROUTINE INTAPE (NTAPE,TAPEID)	0	855
	REAL*8 TAPEID,BUF,EOT	0	856
	CATA 121,BUF,EOT/1.0.0 0.3HEOT/	0	857
	CATA NOT / 6/	0	858
		0	859
C		0	860
C	INITIALIZE TAPE FOR SUBROUTINE WTAPE.	0	861
C	CALLS FORMA SUBROUTINE PAGEHD.	0	862
C	CODED BY RF HRUDA. JULY 1968.	0	863
C	REVISED BY K A PHILIPPUS. APRIL 1969.	0	864
C		0	865
C	SUBROUTINE ARGUMENTS (ALL INPUT)	0	866
C	NTAPE = NUMBER OF TAPE. (E.G. 10).	0	867
C	TAPEID = TAPE IDENTIFICATION. (E.G. T1234). (A6 FORMAT).	0	868
C		0	869
	2001 FORMAT (//// 14H LOGICAL UNIT 12, 7H, TAPE A6,	0	870
	* 23H, HAS BEEN INITIALIZED.)	0	871
C		0	872
	REWIND NTAPE	0	873
	WRITE (NTAPE) TAPEID,121,EOT,(BUF,I=1,16)	0	874
	REWIND NTAPE	0	875
	CALL PAGEHD	0	876
	WRITE (NOT,2001) NTAPE,TAPEID	0	877
C		0	878
	RETURN	0	879
	END	0	880
	SUBROUTINE READ (A,NR,NC,KR,KC)	0	881
C	REAL*8 A,X,IEMRK,ANAME,IZ2,REWIND,LIST,TID,IETCK,EOT	0	882
	DIMENSION A(KR,1),X(4),IEMRK(9)	0	883
	CATA N11,NOT/5,6/	0	884
	CATA REWIND,LIST,EOT,10ULAR,IASTRS/ 6HREWIND,4HLIST,3HEOT,1H\$,1H*/	0	885
		0	886
C		0	887
C	READ MATRIX OF REAL NUMBERS FROM CARDS OR TAPE AND PRINT IT. WRITE	0	888
C	MATRIX ON TAPE IF SO INDICATED (BY HAVING THE WRITE-TAPE NUMBER IN	0	889
C	COLUMNS 79-80).	0	890
C	THE EXPLANATION OF FORMATS USED BELOW IS ...	0	891
C	A - DENOTES ANY KLY PUNCH SYMBOL. (EG, A1/*C).	0	892
C	I - DENOTES AN INTEGER NUMBER. (EG, 436).	0	893
C	E - DENOTES A REAL NUMBER. (EG, 24.963).	0	894
C	**** CARD INPUT ****	0	895
C	FIRST CARD - MATRIX NAME, NUMBER OF ROWS, NUMBER OF COLUMNS	0	896
C	WITH A6,I4,I5 FORMAT.	0	897
C	- REMARKS IN COLUMNS 16-69. A-TYPE FORMAT.	0	898
C	- \$ IN COLUMN 72 FOR WRITE-TAPE INITIALIZATION.	0	899
C	- WRITE-TAPE CONTROL IN COLUMNS 73-78. MAY BE BLANK, OR	0	900
C	THE WORDS REWIND OR LIST, OR (WHEN \$ IN COLUMN 72)	0	901
C	THE WRITE-TAPE-ID (EG, T1234).	0	902
C	- WRITE-TAPE NUMBER IN COLUMNS 79-80. (EG, 21).	0	903
C	MIDDLE CARDS - DATA WITH FORMAT (2I5, 4E17).	0	904
C	- 1-ST I5 IS THE ROW NUMBER.	0	905
C	- 2-ND I5 IS THE COL NUMBER OF THE NEXT E17 FIELD.	0	906
C	- NEXT 4E17 ARE ELEMENTS OF THE MATRIX.	0	907
C	LAST CARD - TEN ZEROS IN COLUMNS 1-10.	0	908

C **** TAPE INPUT ****	0 909
C ONE CARD - MATRIX NAME, ZERO OR MINUS THE LOCATION NUMBER OF MATRIX	0 910
C ON HEAD-TAPE, READ-TAPE NUMBER (IF MINUS, NO PRINTOUT),	0 911
C MATRIX RUN NUMBER WITH A6,I4,I5,A6 FORMAT.	0 912
C - READ-TAPE CONTROL IN COLUMNS 22-27. MAY BE BLANK, OR THE	0 913
C WORDS REWIND OR LIST.	0 914
C - REMARKS IN COLUMNS 28-69. A-TYPE FORMAT.	0 915
C - \$ IN COLUMN 72 FOR WRITE-TAPE INITIALIZATION.	0 916
C - WRITE-TAPE CONTROL IN COLUMNS 73-78. MAY BE BLANK, OR	0 917
C THE WORDS REWIND OR LIST, OR (WHEN \$ IN COLUMN 72)	0 918
C THE WRITE-TAPE-ID (EG, T1234).	0 919
C - WRITE-TAPE NUMBER IN COLUMNS 79-80. (EG, 21).	0 920
C CALLS FERMA SUBROUTINES INTAPE,LTAPE,PAGEHD,RTAPE,WRITE,WTAPE,ZZBOMB.	0 921
C CCDED BY RF HKUDA. JULY 1968.	0 922
C	0 923
C SUBROUTINE ARGUMENTS	0 924
C A = OUTPUT MATRIX READ FROM CARDS OR TAPE.	0 925
C NR = OUTPUT NUMBER OF ROWS IN MATRIX A.	0 926
C NC = OUTPUT NUMBER OF COLS IN MATRIX A.	0 927
C KR = INPUT ROW DIMENSION OF A IN CALLING PROGRAM.	0 928
C KC = INPUT COL DIMENSION OF A IN CALLING PROGRAM.	0 929
C	0 930
1001 FORMAT (A6,I4,I5,9A6, 2X A1,A6,I2)	0 931
1002 FORMAT (2I5,4D17.0)	0 932
2001 FORMAT (//19H CARD INPUT MATRIX A6, 2X 1H(I4,2H X I4,2H)	0 933
* 2X 9A6,2X A1,A6,I4//)	0 934
2002 FORMAT (//19H CARD INPUT MATRIX A6, 2X 1H(I4,2H X I4,2H)	0 935
* 3X 9HCONTINUED //)	0 936
2003 FORMAT (// 1XA6,I4,I5,5X 9A6,2X A1,A6,I4)	0 937
2004 FORMAT (1X 2I5,4D17.8)	0 938
2005 FORMAT (13H\$END OF READ.)	0 939
2006 FORMAT (25H\$SIZE OF MATRIX READ IS (I4,2H X I4,2H))	0 940
C	0 941
C READ IN HEADER CARD.	0 942
READ (NIT,1001) ANAME,N1,N2,I\$REMRK,I\$Z1,I\$Z2,N\$WTAPE	0 943
I\$PRIN = 0	0 944
IF (I\$Z1.EQ.1\$ASTRS .OR. N\$WTAPE.GT.0 .OR. N1.LE.0) I\$PRIN = 1	0 945
IF (I\$PRIN .EQ. 1) CALL PAGEHD	0 946
C	0 947
IF (N1.LE.0) GO TO 200	0 948
C	0 949
C CARD READING SECTION.	0 950
NR = N1	0 951
NC = N2	0 952
IF (I\$PRIN.EQ.1) WRITE (NUT,2001) ANAME,NR,NC,I\$REMRK,I\$Z1,I\$Z2,N\$WTAPE	0 953
N\$ERRCR = 1	0 954
IF (NR.GT.KR .OR. NC.GT.KC) GO TO 999	0 955
NLINE = 0	0 956
DO 105 I=1,NR	0 957
DO 105 J=1,NC	0 958
105 A(I,J) = 0.D 0	0 959
110 READ (NIT,1002) I,J\$X	0 960
IF (I.EQ.0 .AND. J\$.EQ.0) GO TO 300	0 961
N\$ERRCR = 2	0 962
IF (I.LE.0 .OR. I.JT.NR .OR. J\$.LE.0 .OR. J\$.GT.NC) GO TO 999	0 963
JE = J\$+3	0 964
IF (JE.LE.NC) GO TO 115	0 965
JX = NC-J\$+2	0 966
N\$ERRCR = 3	0 967
GO 112 J=JX,4	0 968

112	IF (X(J) .NE. 0.D 0) GO TO 998	0	969
	JE = NC	0	970
115	N = C	0	971
	DO 120 J=JS,JE	0	972
	N = N+1	0	973
120	A(I,J) = X(N)	0	974
	NLINE = NLINE+1	0	975
	IF (NLINE.LE.47) GO TO 125	0	976
	IF (IPRIN .EQ. 1) CALL PAGEHD	0	977
	IF (IPRIN .EQ. 1) WRITE (NCT,2002) ANAME,NR,NC	0	978
	NLINE = 1	0	979
125	IF (IPRIN .EQ. 1) WRITE (NCT,2004) I,JS,(A(I,J),J=JS,JE)	0	980
	GO TO 110	0	981
C		0	982
C	TAPE READING SECTION.	0	983
200	WRITE (NCT,2003) ANAME,N1,N2,IEMRK,I21,I22,NWTAPE	0	984
	NRTAPE = IABS(N2)	0	985
	IF (IEMRK(2) .EQ. REWIND) REWIND NRTAPE	0	986
	IF (IEMRK(2) .EQ. LIST) CALL LTAPE (NRTAPE)	0	987
	IF (N1.EQ.0) GO TO 250	0	988
C	POSITION NRTAPE.	0	989
	READ (NRTAPE) TID,LN,IEUTCK	0	990
	NUM = LN+N1	0	991
	IF (NUM) 205,220,225	0	992
205	NEERRC = 4	0	993
	IF (IEUTCK .EQ. EUT) GO TO 997	0	994
	READ (NRTAPE)	0	995
	NUM = -NUM-1	0	996
	IF (NUM.EQ.0) GO TO 250	0	997
	DO 210 L=1,NUM	0	998
	READ (NRTAPE) TID,LN,IEUTCK	0	999
	NEERRC = 5	0	1000
	IF (IEUTCK .EQ. EUT) GO TO 997	0	1001
210	READ (NRTAPE)	0	1002
	GO TO 250	0	1003
220	EACKSPACE NRTAPE	0	1004
	GO TO 250	0	1005
225	REWIND NRTAPE	0	1006
	NUM = (-N1-1)*2	0	1007
	IF (NUM.EQ.0) GO TO 250	0	1008
	DO 230 L=1,NUM	0	1009
230	READ (NRTAPE)	0	1010
250	CALL RTAPE (IEMRK(1),ANAME,A,NR,NC,KR,KC,NRTAPE)	0	1011
	WRITE (NCT,2006) NR,NC	0	1012
	IF (N2 .GT. 0) CALL WRITE (A,NR,NC,ANAME,KR)	0	1013
C		0	1014
C	TAPE WRITING SECTION.	0	1015
300	IF (NWTAPE.LE.0) GO TO 400	0	1016
	IF (I21 .EQ. IDOLAK) CALL INTAPE (NWTAPE,I22)	0	1017
	IF (I22 .EQ. REWIND) REWIND NWTAPE	0	1018
	CALL RTAPE (A,NR,NC,ANAME,KR,NWTAPE)	0	1019
	IF (I22 .EQ. LIST) CALL LTAPE (NWTAPE)	0	1020
C		0	1021
400	IF (IPRIN .EQ. 1) WRITE (NCT,2005)	0	1022
	RETURN	0	1023
C		0	1024
997	CALL LTAPE (NRTAPE)	0	1025
	GO TO 995	0	1026
998	WRITE (NCT,2004) I,JS,X	0	1027
999	WRITE (NCT,2010) NLEKUR	0	1028

2010 FORMAT (1H1,42HPROGRAM STOPPED. ERROR IN SUBROUTINE READ.	0 1029
* 10H NERROR = ,13)	0 1030
STOP	0 1031
END	0 1032
SUBROUTINE RTAPE (IARUNO,IANAME, A,NRA,NCA, KR,KC,NTAPE)	0 1033
C	0 1034
C	0 1035
REAL*8 A,IARUNO,IANAME,TAPEID,IEOTCK,ITRUNO,ITNAME,	0 1036
* DATE,ITYPE,DENSE,EOT	0 1037
DIMENSION A(KR,1)	0 1038
DATA NUT / 6 /	0 1039
DATA DENSE,EOT / SHDENSE,3HEOT /	0 1040
C	0 1041
C READ MATRIX A FROM TAPE BY IDENTIFICATION OF IARUNO,IANAME.	0 1042
C CALLS FORMA SUBROUTINES LTAPE,PAGEHD,ZZDOMB.	0 1043
C CODED BY WA BENFIELD. JUNE 1966.	0 1044
C LAST REVISION BY R F HRUDA. SEPTEMBER 1971.	0 1045
C	0 1046
C SUBROUTINE ARGUMENTS	0 1047
C IARUNO = INPUT RUN NUMBER OF MATRIX A. (A6 FORMAT).	0 1048
C IANAME = INPUT MATRIX IDENTIFICATION. (A6 FORMAT).	0 1049
C A = OUTPUT MATRIX READ FROM TAPE. SIZE(NRA,NCA).	0 1050
C NRA = OUTPUT NUMBER OF ROWS OF MATRIX A. WILL BE READ FROM TAPE.	0 1051
C NCA = OUTPUT NUMBER OF CCLS OF MATRIX A. WILL BE READ FROM TAPE.	0 1052
C KR = INPUT ROW DIMENSION OF A IN CALLING PROGRAM.	0 1053
C KC = INPUT COL DIMENSION OF A IN CALLING PROGRAM.	0 1054
C NTAPE = INPUT NUMBER OF TAPE. (E.G. 10).	0 1055
C	0 1056
3001 FORMAT (29H1RTAPE CANNOT FIND RUNNO = A6 /	0 1057
* 21X 8HANAME = A6 / 29X 6H-----)	0 1058
C	0 1059
NTIME = 0	0 1060
C SEARCH TAPE FOR CORRECT HEADING.	0 1061
5 READ (NTAPE) TAPEID,LN,IEOTCK,ITRUNO,ITNAME,NRA,NCA,DATE,ITYPE,NNZ	0 1062
IF (ITRUNO.EQ. IARUNO .AND. ITNAME.EQ. IANAME) GO TO 10	0 1063
IF (IEOTCK.EQ. EOT) GO TO 20	0 1064
READ (NTAPE)	0 1065
GO TO 5	0 1066
C	0 1067
C MATRIX HAS BEEN FOUND.	0 1068
10	0 1069
IF (ITYPE.NE. DENSE .AND. NNZ.NE. 0) GO TO 999	0 1070
	0 1071
IF (NRA.GT.KR .OR. NCA.GT.KC) GO TO 999	0 1072
READ (NTAPE) ((A(I,J),I=1,NRA),J=1,NCA)	0 1073
RETURN	0 1074
C	0 1075
C MATRIX CANNOT BE FOUND. SEARCH TAPE ONCE MORE.	0 1076
20 NTIME = NTIME+1	0 1077
	0 1078
IF (NTIME.EQ. 2) GO TO 998	0 1079
REWIND NTAPE	0 1080
GO TO 5	0 1081
C	0 1082

998 WRITE (NCT,3001) IARUND,IANAME	C 1083
999 CALL LTAPE (INTAPE)	O 1084
WRITE (NCT,3002) NERROR	O 1085
3002 FORMAT (1H1,43HERROR IN SUBROUTINE RTAPE, PROGRAM STOPPED,	O 1086
* 10H NERROR = ,13)	O 1087
STOP	O 1088
END	O 1089

SUBROUTINE WRITE (A,NR,NC,ANAME,KR)	O 1090
C	DEBUG # 12 O 1091
REAL*8 A,ANAME	O 1092
DIMENSION A(KR,1)	O 1093
DATA NOT / 6/	O 1094
C	O 1095
C WRITE MATRIX OF REAL NUMBERS ON PAPER.	O 1096
C REQUIRES 123 COLUMN (MINIMUM) PRINTER.	O 1097
C UP TO 10 DATA FIELDS PER LINE. PRINTS ONLY NON-ZERO FIELD ROWS.	O 1098
C CALLS FORMA SUBROUTINE PAGEHD.	O 1099
C CODED BY RL WOHLER. DECEMBER 1968.	O 1100
C	O 1101
C SUBROUTINE ARGUMENTS (ALL INPUT)	O 1102
C A = MATRIX TO BE PRINTED. SIZE(NR,NC).	O 1103
C NR = NUMBER OF ROWS IN MATRIX A.	O 1104
C NC = NUMBER OF COLS IN MATRIX A.	O 1105
C ANAME = MATRIX IDENTIFICATION. (A6 FORMAT).	O 1106
C KR = ROW DIMENSION OF A IN CALLING PROGRAM.	O 1107
C	O 1108
2010 FORMAT (//15H OUTPUT MATRIX A6,2X 1H(14.2H X 14.2H) //	O 1109
* 10X,10(7X,1H(12,1H))//)	O 1110
2020 FORMAT (//15H OUTPUT MATRIX A6,2X 1H(14.2H X 14.2H)	O 1111
* 3X, 9HCONTINUED //10X,10(7X,1H(12,1H))//)	O 1112
2030 FORMAT (1X,215,2X,1P10D11.3)	O 1113
2040 FORMAT (14H0END OF WRITE.)	O 1114
C	O 1115
C PULL UP A NEW PAGE FOR MATRIX AND PRINT MATRIX NAME.	O 1116
CALL PAGEHD	O 1117
WRITE (NOT,2010) ANAME,NR,NC,(L,L=1,10)	O 1118
NLINE = 0	O 1119
C	O 1120
DO 60 I=1,NR	O 1121
NZERC = 0	O 1122
JS = 1	O 1123
10 JE = JS+9	O 1124
IF (JE .GT. NC) JE=NC	O 1125
C SEE IF ELEMENTS ARE ZERO.	O 1126
DO 20 J=JS,JE	O 1127
20 IF (A(I,J) .NE. 0.0 0) GO TO 30	O 1128
GO TO 40	O 1129
30 NLINE = NLINE+1	O 1130
IF (NLINE .LE. 44) GO TO 35	O 1131
CALL PAGEHD	O 1132
WRITE (NCT,2020) ANAME,NR,NC,(L,L=1,10)	O 1133
NLINE = 1	O 1134
35 WRITE (NCT,2030) I,JS,(A(I,J), J=JS,JE)	O 1135
NZERC = 1	O 1136

40 IF (JE .EQ. NC) GO TO 50	0 1137
JS = JS+10	0 1138
GO TO 10	0 1139
C SKIP A SPACE BETWEEN EACH ROW IF THERE ARE MORE THAN 10 COLUMNS	0 1140
C AND SOMETHING HAS BEEN WRITTEN.	0 1141
50 IF (NC.LE.10 .OR. NZERO.EQ.0 .OR. I.EQ.NR) GO TO 60	0 1142
NLINE = NLINE+1	0 1143
WRITE (NCT,2030)	0 1144
60 CONTINUE	0 1145
C	0 1146
WRITE (NCT,2040)	0 1147
RETURN	0 1148
END	0 1149
SUBROUTINE WTAPE (A,NRA,NCA,ANAME,KR,NTAPE)	0 1150
C	0 1151
REAL*8 A,ANAME,IRUNNO,DATE,UNAME,TITLE1,TITLE2,	0 1152
* BUF,EUT,DENSE,TAPEID,IEOTCK	0 1153
DIMENSION A(KR,1)	0 1154
COMMON /LSTART/IRUNNO,DATE,NPAGE	0 1155
COMMON /LSTRT1/ UNAME(3),TITLE1(12),TITLE2(12)	0 1156
DATA BUF,EUT,DENSE/0.0 0.3HEUT,SHDENSE/	0 1157
DATA NUT / 0/	0 1158
C	0 1159
C WRITE MATRIX A ON TAPE.	0 1160
C INITIALIZE TAPE WITH SUBROUTINE INTAPE.	0 1161
C REWIND TAPE BEFORE FIRST USE OF THIS SUBROUTINE.	0 1162
C NOTE...THIS ROUTINE IS DESIGNED SPECIFICALLY FOR WRITING ON A DISK	0 1163
C	0 1164
C (EG CDC-6400 DISK). USING THIS ROUTINE TO WRITE ON A PHYSICAL	0 1165
C TAPE DIRECTLY (IE WITHOUT USING THE DISK AS AN INTERMEDIARY)	0 1166
C WILL PROBABLY GIVE POOR RESULTS (DUE TO THE TOLERANCE	0 1167
C CHARACTERISTICS OF A TAPE DRIVE) AND SHOULD BE AVOIDED IF AT	0 1168
C ALL POSSIBLE.	0 1169
C ...THE CDC-6400 DISK IS AUTOMATICALLY ENDFILED AFTER EACH WRITE.	0 1170
C CODED BY W A BENFIELD. MARCH 1966.	0 1171
C REVISED BY R A PHILIPPUS. APRIL 1969.	0 1172
C REVISED BY RF HRUDA. NOVEMBER 1970.	0 1173
C	0 1174
C SUBROUTINE ARGUMENTS (ALL INPUT)	0 1175
C A = MATRIX TO BE WRITTEN ON TAPE. SIZE(NRA,NCA).	0 1176
C NRA = NUMBER OF ROWS OF MATRIX A.	0 1177
C NCA = NUMBER OF COLS OF MATRIX A.	0 1178
C ANAME = MATRIX IDENTIFICATION. (A6 FORMAT).	0 1179
C KR = ROW DIMENSION OF A IN CALLING PROGRAM.	0 1180
C NTAPE = NUMBER OF TAPE. (E.G. 10).	0 1181
C	0 1182
C INTERNAL VARIABLES THAT ARE PUT ON TAPE (TRANSFERRED THRU COMMON).	0 1183
C IRUNNO IS RUN NUMBER OF PROBLEM. (A6 FORMAT).	0 1184
C DATE IS DATE. (A6 FORMAT). FOR EXAMPLE 15FEB69.	0 1185
C	0 1186
IF (NRA .LT. 1 .OR. NCA .LT. 1) GO TO 999	0 1187
C	0 1188
C SEARCH TAPE FOR END OF WRITTEN DATA.	0 1189
10 READ (NTAPE) TAPEID,LN,IEOTCK	0 1190

IF (IEOTCK .EQ. EOT) GO TO 20	0 1191
READ (NTAPE)	0 1192
GO TO 10	0 1193
C	0 1194
C END OF WRITTEN DATA HAS BEEN FOUND.	0 1195
20 EACKSPACE NTAPE	0 1196
WRITE (NTAPE) TAPEID, LN, BUF, IRUNNO, ANAME, NRA, NCA, DATE, DENSE,	0 1197
* (BUF, I=1, 10)	0 1198
WRITE (NTAPE) ((A(I, J), I=1, NRA), J=1, NCA)	0 1199
LN = LN + 1	0 1200
WRITE (NTAPE) TAPEID, LN, EOT, (BUF, I=1, 16)	0 1201
EACKSPACE NTAPE	0 1202
RETURN	0 1203
C	0 1204
999 WRITE (NCT, 1000)	0 1205
1000 FORMAT (1H1, 43HERROR IN SUBROUTINE WTAPE. PROGRAM STOPPED.)	0 1206
STOP	0 1207
END	0 1208
SUBROUTINE LTAPE (NTAPE)	0 1209
C	0 1210
REAL*8 TAPEID, IRUNNO, ANAME, IEOTCK, DATE, ITYPE, ICHK, EOT,	0 1211
* DENSE, SPARSE, SPART	0 1212
DATA NOT / 6/	0 1213
DATA EOT, DENSE, SPARSE, SPART/ 3HEOT, 5HDENSE, 6HSPARSE, 5HSPART/	0 1214
C	0 1215
C LIST HEADINGS OF MATRICES ON TAPE.	0 1216
C CALLS FERMA SUBROUTINE PAGEHD.	0 1217
C CODED BY RF HRUDA. JULY 1968. REVISED NOVEMBER 1970.	0 1218
C REVISED BY R A PHILIPPUS. APRIL 1969.	0 1219
C	0 1220
C SUBROUTINE ARGUMENTS (ALL INPUT)	0 1221
C NTAPE = NUMBER OF TAPE. (E.G. 10).	0 1222
C	0 1223
2001 FORMAT (/36X35HLISTING OF MATRICES ON LOGICAL UNIT13, 7H, TAPE A6)	0 1224
2002 FORMAT (/30X35HLISTING OF MATRICES ON LOGICAL UNIT13, 7H, TAPE A6,	0 1225
* 12H (CONTINUED))	0 1226
2003 FORMAT (27X69(1H-)/27X3HNO.3X7HRUN NO.4X4HNAME5X3HNROWS4X5HNCOLS4X	0 1227
* 4HDATE6X3HWNZ3X9HPARTITION/	0 1228
* 27X3H---3X6H-----4X6H-----4X5H-----	0 1229
* 4X5H-----3X6H-----5X3H---3X9H----- /)	0 1230
2004 FORMAT (25X15, 3XA6, 4XA6, 3X15, 4X15, 4XA6, 3X15, 3X14, 1H/14)	0 1231
2005 FORMAT (/27X12HEND OF LIST.)	0 1232
C	0 1233
REWIND NTAPE	0 1234
READ (NTAPE) TAPEID	0 1235
REWIND NTAPE	0 1236
L=C	0 1237
C	0 1238
12 CALL PAGEHD	0 1239
IF (L .EQ. 0) WRITE (NCT, 2001) NTAPE, TAPEID	0 1240
IF (L .NE. 0) WRITE (NCT, 2002) NTAPE, TAPEID	0 1241
WRITE (NCT, 2003)	0 1242
NLINE=1	0 1243
13 L=L+1	0 1244

	READ (NTAPE) TAPEID, LN, IEUTCK, IRUNNC, ANAME, NR, NC, DATE, ITYPE, NNZ,	0 1245
	* MP, NPT	0 1246
	IF (L.EQ. 1) ICHK = IRUNNC	0 1247
	IF (ICLK.EQ. IRUNNC) GO TO 15	0 1248
	NLINE=NLINE+1	0 1249
	WRITE (NOT, 2004)	0 1250
	ICLK = IRUNNC	0 1251
15	IF (IEUTCK.EQ. EOT) GO TO 30	0 1252
	READ (NTAPE)	0 1253
	IF (ITYPE.EQ. DENSE) WRITE (NOT, 2004)	0 1254
	* LN, IRUNNO, ANAME, NR, NC, DATE	0 1255
	IF (ITYPE.EQ. DENSE) GO TO 20	0 1256
	IF (ITYPE.EQ. SPARSE) WRITE (NOT, 2004)	0 1257
	* LN, IRUNNO, ANAME, NR, NC, DATE, NNZ	0 1258
	IF (ITYPE.EQ. SPARSE) GO TO 20	0 1259
	IF (ITYPE.EQ. SPART) WRITE (NOT, 2004)	0 1260
	* LN, IRUNNO, ANAME, NR, NC, DATE, NNZ, NP, NPT	0 1261
	IF (ITYPE.EQ. SPART) GO TO 20	0 1262
	WRITE (NOT, 2004) LN, IRUNNO, ANAME, NR, NC, ITYPE	0 1263
20	NLINE=NLINE+1	0 1264
	IF (NLINE.GT. 43) GO TO 12	0 1265
	GO TO 13	0 1266
C		0 1267
30	WRITE (NOT, 2004) LN, IEUTCK	0 1268
	WRITE (NOT, 2005)	0 1269
	REWIND NTAPE	0 1270
	RETURN	0 1271
	END	0 1272
	SUBROUTINE MSMODC (NBOD)	0 1273
C		0 1274
	IMPLICIT REAL*8(A-H, O-Z)	0 1275
C		0 1276
	COMMON /MAXIMUM/	0 1277
	* N3MAX, NHMAX, NSPMAX, NMWMAX, NMWBOD, NMDODD, KMJ, KY, KU	0 1278
	COMMON /NHNS /	0 1279
	* NHPC(6), NSPG(6)	12 1280
	COMMON /NUMBRS/	0 1281
	* ZRC, ONE, TWO, THES	0 1282
	COMMON /SPECIF/	0 1283
	* BETAH(6, 6), DELTAH(6, 6), AMD(2, 5), RH(3, 3, 30), RS(3, 3, 30),	16 1284
	* CH(3, 35), DS(3, 30), IMU(3, 5), NMOW(6, 6), IFTSMW(15),	17 1285
	* N3, NH, NSPT, NOFMU, NDELTA, ITCPOL(2, 6), IRGFLX(6), IHDATA(7, 6),	18 1286
	* LCCU(14), LENU(14), NO, NBETA, NLAM, NEQ	19 1287
	COMMON /SUMMRY/	0 1288
	* ASUMRY(15, 6), ISUMRY(15, 3), KSUMRY	98 1289
	COMMON /TAPENG/	0 1290
	* NTAPE1, NTAPE2, NTAPE3	0 1291
C		0 1292
	DIMENSION A(42, 42), B(42, 42), IV(42), JV(42), C(6, 6), PHR6(6, 6),	61 1293
	* BC(9, 12), WS1(12, 12), WS2(12, 12), CM2(42), UMGA2(18), JDOF(7, 6)	62 1294
C		0 1295
	DATA NIT, NOT, KAB, KJDOF/ 5, 6, 42, 7 /	63 1296
	1001 FORMAT (1615)	0 1297
	1002 FORMAT (8D10.3)	0 1298

3001	FORMAT (//15X30HSUMMARY OF INPLT DATA FOR BODY,13,	0 1299
	* 44H WHICH IS FLEXIBLE W/CONSISTENT MASS MATRIX.//	0 1300
	* 3X4\$H THE INTEGER PARAMETERS--- IFRBM,IDIAM,IDIAD ARE\$X12,1H,	0 1301
	* 5X12,1H,5X12,//	0 1302
	* 3X25H THE JOOF TABLE FOLLOWS---	0 1303
3002	FORMAT (//3X36H THE MODE SELECTION VECTOR FOLLOWS---	0 1304
3003	FORMAT (//3X12H FOR BODY NO.,13,25H THE POSITION VECTOR FROM	0 1305
	* 25H THE BODY ORIGIN TO JOINT,14,3H IS. /	0 1306
	* 10X 4HX = 1PD10.3,5X5H Y = 1PD10.3,5X5H Z = 1PD10.3)	0 1307
3004	FORMAT (//5X46H THE CONSISTENT, REPARTITIONED MASS MATRIX IS--)	0 1308
3005	FORMAT (//5X36H THE REPARTITIONED MODAL MATRIX IS---	0 1309
3006	FORMAT (//5X44H THE -UNDEFORMED- INERTIA MATRIX (MG) IS---	0 1310
3007	FORMAT (//5X27H THE A COEFFICIENTS ARE---	0 1311
3008	FORMAT (//5X27H THE B COEFFICIENTS ARE---	0 1312
3009	FORMAT (//5X31H THE COFX1 COEFFICIENTS ARE---	0 1313
3010	FORMAT (//5X31H THE COFX2 COEFFICIENTS ARE---	0 1314
3011	FORMAT (//5X31H THE COFYZ COEFFICIENTS ARE---	0 1315
3012	FORMAT (//5X31H THE C11 COEFFICIENTS ARE---	0 1316
3013	FORMAT (//5X31H THE C22 COEFFICIENTS ARE---	0 1317
3014	FORMAT (//5X31H THE C33 COEFFICIENTS ARE---	0 1318
3015	FORMAT (//5X31H THE C12 COEFFICIENTS ARE---	0 1319
3016	FORMAT (//5X31H THE C13 COEFFICIENTS ARE---	0 1320
3017	FORMAT (//5X31H THE C23 COEFFICIENTS ARE---	0 1321
3018	FORMAT (//5X33H THE MODAL STIFFNESS MATRIX IS---	0 1322
3019	FORMAT (//5X31H THE MODAL DAMPING MATRIX IS---	0 1323
3020	FORMAT (//5X50H THE INITIAL MODAL COORDINATE DISPLACEMENTS ARE---	0 1324
3021	FORMAT (//5X47H THE INITIAL MODAL COORDINATE VELOCITIES ARE---	0 1325
3022	FORMAT (//5X 9HFOR BODY 13,29H THE P-Q HINGE NO., THE EULER	0 1326
	* 57H ROTATION TYPE AND THE JOINT NO. CORRESPONDING TO THE P-Q, /	0 1327
	* 5X54H HINGE APPEAR IN THE FOLLOWING INTEGER ARRAY WHICH IS	0 1328
	* 49HFOLLOWED BY AN ARRAY CONTAINING EULER ANGLES THAT, /	0 1329
	* 5X44H POSITION THE HINGE TRIAD WRT THE BODY TRIAD)	0 1330
3023	FORMAT (//5X 9HFOR BODY 13,32H THE SENSOR POINT NO., THE EULER	0 1331
	* 60H ROTATION TYPE AND THE JOINT NO. CORRESPONDING TO THE SENSOR, /	0 1332
	* 6X53H POINT APPEAR IN THE FOLLOWING INTEGER ARRAY WHICH IS	0 1333
	* 49HFOLLOWED BY AN ARRAY CONTAINING EULER ANGLES THAT, /	0 1334
	* 5X45H POSITION THE SENSOR TRIAD WRT THE BODY TRIAD)	0 1335
C		0 1336
	REWIND NTAPE2	0 1337
	KMC = NMODOD	0 1338
	CALL ZERO (BC,9,KMO,9)	0 1339
	READ (NIT,1001) IFRBM,IDIAM,IDIAD	0 1340
	CALL READIM (JOOF,NX,NO,KJOOF,6)	0 1341
	IC = 0	0 1342
	DO 2 J=1,6	0 1343
	DO 2 I=1,NX	0 1344
	NCF = JOCF(1,J)	0 1345
	IC = IC + 1	0 1346
2	IV(NCF) = IC	0 1347
	CALL READIM (JV,N1,N2,1,KAB)	0 1348
	NY = NX	0 1349
	NZ = NX	0 1350
	NMC = 0	0 1351
	CALL PAGEHD	0 1352
	WRITE (NOT,3001) NMOD,IFRBM,IDIAM,IDIAD	0 1353
	CALL WRITIS (JOOF,NX,6,KJOOF)	0 1354
	WRITE (NGT,3002)	0 1355
	CALL WRITIS (JV,1,N2,1)	0 1356
C		0 1357
	DO 3 I=1,N2	0 1358

IF (JV(1) .EQ. 0) GO TO 3	0 1359
NMO = NMO + 1	0 1360
3 CONTINUE	0 1361
IF (NMO .GT. KMO + 6) GO TO 999	0 1362
CALL READ (A,NRA,NCA,KAB,KAB)	0 1363
CALL REVISE (A,IV,IV,B,NRA,NCA,NRA,NRA,KAB,KAB)	0 1364
WRITE (NTAPE2) ((B(I,J),I=1,NRA),J=1,NRA)	0 1365
REWIND NTAPE2	0 1366
C	0 1367
CALL READ (A,NRA,NMOT,KAB,KAB)	0 1368
IF (IDIAK .EQ. 0 .AND. IDIAD .EQ. 0) GO TO 11	0 1369
CALL READ (OM2,N1,N2,1,KAB)	0 1370
11 NE = NMO - 6	0 1371
CALL REVISE (A,IV,JV,B,NRA,NMOT,NRA,NMO,KAB,KAB)	0 1372
IF (IDIAK .EQ. 0 .AND. IDIAD .EQ. 0) GO TO 12	0 1373
CALL REVISE (OM2,1,JV,OMGA2,1,N2,1,NMO,1,1)	0 1374
12 IF (IFREM .EQ. 0) GO TO 5	0 1375
READ (NIT,1001) JTYPC	0 1376
READ (NIT,1002) (OM2(J),J=1,3)	0 1377
WRITE (NCT,3003) NEUD,JTYPC,OM2(1),OM2(2),OM2(3)	0 1378
JRB = JTYPC - NX	0 1379
DO 4 I=1,6	0 1380
JRB = JRB + NX	0 1381
DO 4 J=1,6	0 1382
4 PHR6(I,J) = B(JRB,J)	0 1383
CALL GACSS1 (PHR6,C,6,6)	0 1384
CALL ZENC (PHR6(4,4),3,3,6)	0 1385
CALL UNITY (PHR6(1,4),3,6)	0 1386
CALL UNITY (PHR6(4,1),3,6)	0 1387
CALL SKEW3 (OM2,PHR6,1,6)	0 1388
CALL MULTA (C,PHR6,6,6,6,6,6)	0 1389
CALL MULTA (B,C,NKA,6,6,KAB,6)	0 1390
C	0 1391
5 READ (NTAPE2) ((A(I,J),I=1,NRA),J=1,NRA)	0 1392
REWIND NTAPE2	0 1393
WRITE (NCT,3004)	0 1394
CALL WRITES (A ,NRA,NRA,KAB)	0 1395
WRITE (NCT,3005)	0 1396
CALL WRITES (B ,NKA,NMO,KAB)	0 1397
C	0 1398
CALL BTABA (A,B,NRA,NMO,KAB,KAB)	0 1399
WRITE (NCT,3006)	0 1400
CALL WRITES (A ,NMO,NMC,KAB)	0 1401
WRITE (NTAPE1) ((A(I,J),J=1,NMO),I=1,NMO)	0 1402
DO 25 J=1,NE	0 1403
JP6 = J + 6	0 1404
25 CM2(J) = A(JP6,JP6)	0 1405
C	0 1406
READ (NTAPE2) ((A(I,J),I=1,NRA),J=1,NRA)	0 1407
REWIND NTAPE2	0 1408
NRP = J*NX	0 1409
CALL MULTA (A,B,NRP,NRA,NMC,KAB,KAB)	0 1410
CALL ZLRL (BC,9,NE,9)	0 1411
DO 15 J=1,NE	0 1412
K = 6 + J	0 1413
DO 15 IX = 1,NX	0 1414
IY = IX + NX	0 1415
IZ = IY + NX	0 1416
BC(1,J) = BC(1,J) + A(IZ,4)*B(IY,K) - A(IY,4)*B(IZ,K)	0 1417
BC(2,J) = BC(2,J) + A(IZ,5)*B(IY,K) - A(IY,5)*B(IZ,K)	0 1418

BC(3,J) = BC(3,J) + A(IZ,6)*B(IY,K) - A(IY,6)*B(IZ,K)	0 1419
EC(4,J) = BC(4,J) + A(IX,4)*B(IZ,K) - A(IZ,4)*B(IX,K)	0 1420
EC(5,J) = BC(5,J) + A(IX,5)*B(IZ,K) - A(IZ,5)*B(IX,K)	0 1421
EC(6,J) = BC(6,J) + A(IX,6)*B(IZ,K) - A(IZ,6)*B(IX,K)	0 1422
BC(7,J) = BC(7,J) + A(IY,4)*B(IX,K) - A(IX,4)*B(IY,K)	0 1423
EC(8,J) = BC(8,J) + A(IY,5)*B(IX,K) - A(IX,5)*B(IY,K)	0 1424
EC(9,J) = BC(9,J) + A(IY,6)*B(IX,K) - A(IX,6)*B(IY,K)	0 1425
15 CONTINUE	0 1426
WRITE (NTAPE1) ((BC(I,J),J=1,NE),I=1,9)	0 1427
WRITE (NCT,3007)	0 1428
CALL WRITES (BC , 9,NE , 9)	0 1429
C	0 1430
CALL ZERG (BC,9,NE,9)	0 1431
DO 16 J=1,NE	0 1432
K = 6 + J	0 1433
DO 16 IX=1,NX	0 1434
IY = IX + NX	0 1435
IZ = IY + NX	0 1436
EC(1,J) = BC(1,J) + A(IZ,1)*B(IY,K) - A(IY,1)*B(IZ,K)	0 1437
EC(2,J) = BC(2,J) + A(IX,2)*B(IZ,K) - A(IZ,2)*B(IX,K)	0 1438
EC(3,J) = BC(3,J) + A(IY,3)*B(IX,K) - A(IX,3)*B(IY,K)	0 1439
EC(4,J) = BC(4,J) + A(IZ,1)*B(IX,K) - A(IX,1)*B(IZ,K)	0 1440
* + A(IY,2)*B(IZ,K) - A(IZ,2)*B(IY,K)	0 1441
BC(5,J) = BC(5,J) + A(IX,1)*B(IY,K) - A(IY,1)*B(IX,K)	0 1442
* + A(IY,3)*B(IZ,K) - A(IZ,3)*B(IY,K)	0 1443
EC(6,J) = BC(6,J) + A(IX,2)*B(IY,K) - A(IY,2)*B(IX,K)	0 1444
* + A(IZ,3)*B(IX,K) - A(IX,3)*B(IZ,K)	0 1445
16 CONTINUE	0 1446
WRITE (NTAPE1) ((BC(I,J),J=1,NE),I=1,6)	0 1447
WRITE (NCT,3008)	0 1448
CALL WRITES (BC , 6,NE , 9)	0 1449
C	0 1450
CALL ZERG (WS1,NE,NE,KMG)	0 1451
DO 17 I=1,NE	0 1452
KI = 6 + I	0 1453
DO 17 J=1,NE	0 1454
KJ = 6 + J	0 1455
DO 17 IX=1,NX	0 1456
IY = IX + NX	0 1457
IZ = IY + NX	C 1458
WS1(I,J) = WS1(I,J) + B(IX,KI)*A(IY,KJ) - B(IY,KI)*A(IX,KJ)	0 1459
17 CONTINUE	0 1460
WRITE (NTAPE1) ((WS1(I,J),J=1,NE),I=1,NE)	0 1461
WRITE (NCT,3009)	0 1462
CALL WRITES (WS1,NE ,NE ,KMG)	C 1463
C	0 1464
CALL ZERG (WS1,NE,NE,KMG)	0 1465
DO 18 I=1,NE	0 1466
KI = 6 + I	0 1467
DO 18 J=1,NE	0 1468
KJ = 6 + J	0 1469
DO 18 IX=1,NX	0 1470
IY = IX + NX	0 1471
IZ = IY + NX	C 1472
WS1(I,J) = WS1(I,J) + B(IZ,KI)*A(IX,KJ) - B(IX,KI)*A(IZ,KJ)	0 1473
18 CONTINUE	C 1474
WRITE (NTAPE1) ((WS1(I,J),J=1,NE),I=1,NE)	0 1475
WRITE (NCT,3010)	0 1476
CALL WRITES (WS1,NE ,NE ,KMG)	0 1477
C	0 1478

CALL ZERC (WS1,NE,NE,KMU)	0 1479
DO 15 I=1,NE	0 1480
KI = 6 + I	C 1481
DO 15 J=1,NE	0 1482
KJ = 6 + J	0 1483
DO 15 IX=1,NX	0 1484
IY = IX + NX	0 1485
IZ = IY + NX	C 1486
WS1(I,J) = WS1(I,J) + B(IY,KI)*A(IZ,KJ) - B(IZ,KI)*A(IY,KJ)	0 1487
15 CONTINUE	0 1488
WRITE (NTAPE1) ((WS1(I,J),J=1,NE),I=1,NE)	0 1489
WRITE (NCT,3011)	0 1490
CALL WRITES (WS1,NE,NE,KMU)	0 1491
C	0 1492
LX = 1	0 1493
LY = LX + NX	0 1494
LZ = LY + NY	0 1495
SI = ONE	C 1496
C	0 1497
READ (NTAPE2) ((A(I,J),I=1,NKA),J=1,NRA)	0 1498
C	C 1499
C	C 1500
CALL ZERC (WS1,NE,NE,KMU)	0 1501
CALL PR3 (A(LZ,LZ),B(LY,7),B(LY,7),WS2,WS1,SI,NZ,NZ,NE,NE,	0 1502
* KAE,KAE,KAB,KJDOF,KMU)	0 1503
CALL PR3 (A(LZ,LY),B(LZ,7),B(LY,7),WS2,WS1,-SI,NZ,NY,NE,NE,	0 1504
* KAE,KAE,KAB,KJDOF,KMU)	0 1505
CALL PR3 (A(LY,LY),B(LY,7),B(LZ,7),WS2,WS1,-SI,NY,NZ,NE,NE,	0 1506
* KAE,KAE,KAB,KJDOF,KMU)	0 1507
CALL PR3 (A(LY,LZ),B(LZ,7),B(LZ,7),WS2,WS1,SI,NY,NY,NE,NE,	0 1508
* KAE,KAE,KAB,KJDOF,KMU)	C 1509
WRITE (NTAPE1) ((WS1(I,J),J=1,NE),I=1,NE)	C 1510
WRITE (NCT,3012)	0 1511
CALL WRITES (WS1,NE,NE,KMU)	C 1512
CALL ZERC (WS1,NE,NE,KMU)	C 1513
CALL PR3 (A(LX,LX),B(LZ,7),B(LZ,7),WS2,WS1,SI,NX,NX,NE,NE,	0 1514
* KAE,KAE,KAB,KJDOF,KMU)	0 1515
CALL PR3 (A(LX,LZ),B(LX,7),B(LZ,7),WS2,WS1,-SI,NX,NZ,NE,NE,	0 1516
* KAE,KAE,KAB,KJDOF,KMU)	0 1517
CALL PR3 (A(LZ,LX),B(LZ,7),B(LX,7),WS2,WS1,-SI,NZ,NX,NE,NE,	0 1518
* KAE,KAE,KAB,KJDOF,KMU)	0 1519
CALL PR3 (A(LZ,LZ),B(LX,7),B(LX,7),WS2,WS1,SI,NZ,NZ,NE,NE,	0 1520
* KAE,KAB,KAB,KJDOF,KMU)	0 1521
WRITE (NTAPE1) ((WS1(I,J),J=1,NE),I=1,NE)	0 1522
WRITE (NCT,3013)	0 1523
CALL WRITES (WS1,NE,NE,KMU)	C 1524
CALL ZERO (WS1,NE,NE,KMU)	0 1525
CALL PR3 (A(LY,LY),B(LX,7),B(LX,7),WS2,WS1,SI,NY,NY,NE,NE,	C 1526
* KAE,KAE,KAB,KJDOF,KMU)	0 1527
CALL PR3 (A(LY,LX),B(LY,7),B(LX,7),WS2,WS1,-SI,NY,NX,NE,NE,	0 1528
* KAE,KAE,KAB,KJDOF,KMU)	0 1529
CALL PR3 (A(LX,LY),B(LX,7),B(LY,7),WS2,WS1,-SI,NX,NY,NE,NE,	0 1530
* KAE,KAE,KAB,KJDOF,KMU)	0 1531
CALL PR3 (A(LX,LX),B(LY,7),B(LY,7),WS2,WS1,SI,NX,NX,NE,NE,	0 1532
* KAE,KAE,KAB,KJDOF,KMU)	0 1533
WRITE (NTAPE1) ((WS1(I,J),J=1,NE),I=1,NE)	0 1534
WRITE (NCT,3014)	0 1535
CALL WRITES (WS1,NE,NE,KMU)	0 1536
CALL ZERC (WS1,NE,NE,KMU)	0 1537
CALL PR3 (A(LZ,LX),B(LZ,7),B(LY,7),WS2,WS1,-SI,NZ,NX,NE,NE,	0 1538

* KAE,KAB,KAB,KJDOF,KMO)	0 1539
CALL PR3 (A(LZ,LZ),B(LX,7),B(LY,7),WS2,WS1, S1,NZ,NZ,NE,NE,	0 1540
* KAE,KAE,KAB,KJDOF,KMG)	0 1541
CALL PR3 (A(LY,LX),B(LX,7),B(LZ,7),WS2,WS1, S1,NY,NX,NE,NE,	0 1542
* KAE,KAB,KAB,KJDOF,KMO)	0 1543
CALL PR3 (A(LY,LZ),B(LX,7),B(LZ,7),WS2,WS1,-S1,NY,NZ,NE,NE,	0 1544
* KAE,KAE,KAB,KJDOF,KMO)	0 1545
WRITE (NTAPE1) ((WS1(I,J),J=1,NE),I=1,NE)	0 1546
WRITE (NCT,3015)	0 1547
CALL WRITES (WS1,NE ,NE ,KMO)	0 1548
CALL ZERC (WS1,NE,NE,KMO)	0 1549
CALL PR3 (A(LX,LY),B(LX,7),B(LY,7),WS2,WS1,-S1,NZ,NY,NE,NE,	0 1550
* KAE,KAB,KAB,KJDOF,KMG)	0 1551
CALL PR3 (A(LZ,LX),B(LY,7),B(LY,7),WS2,WS1, S1,NZ,NX,NE,NE,	0 1552
* KAE,KAE,KAB,KJDOF,KMO)	0 1553
CALL PR3 (A(LX,LY),B(LX,7),B(LZ,7),WS2,WS1, S1,NY,NY,NE,NE,	0 1554
* KAE,KAB,KAB,KJDOF,KMO)	0 1555
CALL PR3 (A(LY,LX),B(LY,7),B(LZ,7),WS2,WS1,-S1,NY,NX,NE,NE,	0 1556
* KAE,KAE,KAB,KJDOF,KMO)	0 1557
WRITE (NTAPE1) ((WS1(I,J),J=1,NE),I=1,NE)	0 1558
WRITE (NCT,3016)	0 1559
CALL WRITES (WS1,NE ,NE ,KMG)	0 1560
CALL ZERG (WS1,NE,NE,KMO)	0 1561
CALL PR3 (A(LX,LY),B(LX,7),B(LZ,7),WS2,WS1,-S1,NX,NY,NE,NE,	0 1562
* KAE,KAE,KAB,KJDOF,KMG)	0 1563
CALL PR3 (A(LX,LX),B(LY,7),B(LZ,7),WS2,WS1, S1,NX,NX,NE,NE,	0 1564
* KAE,KAB,KAB,KJDOF,KMO)	0 1565
CALL PR3 (A(LZ,LY),B(LX,7),B(LX,7),WS2,WS1, S1,NZ,NY,NE,NE,	0 1566
* KAE,KAB,KAB,KJDOF,KMG)	0 1567
CALL PR3 (A(LZ,LX),B(LY,7),B(LX,7),WS2,WS1,-S1,NZ,NX,NE,NE,	0 1568
* KAE,KAB,KAB,KJDOF,KMO)	0 1569
WRITE (NTAPE1) ((WS1(I,J),J=1,NE),I=1,NE)	0 1570
WRITE (NCT,3017)	0 1571
CALL WRITES (WS1,NL ,NE ,KMO)	0 1572
C	0 1573
IF (IDIAK .EQ. 1) GO TO 50	0 1574
CALL READ (A,NRA,NCA,KAB,KAB)	0 1575
CALL BTABA (A,B(1,7),NRA,NE,KAB,KAB)	0 1576
GO TO 51	0 1577
50 CALL ZERG (A,NE,NE,KAB)	0 1578
DO 55 J=1,NE	0 1579
JP6 = J + 6	0 1580
55 A(J,J) = UM2(J)*CMGA2(JP6)	0 1581
51 WRITE (NTAPE1) ((A(I,J),J=1,NE),I=1,NE)	0 1582
WRITE (NCT,3018)	0 1583
CALL WRITES (A ,NE ,NE ,KAB)	0 1584
IF (IDIAK .EQ. 1) GO TO 60	0 1585
CALL READ (A,NRA,NCA,KAB,KAB)	0 1586
CALL BTABA (A,B(1,7),NRA,NE,KAB,KAB)	0 1587
GO TO 61	0 1588
60 CALL ZERG (A,NE,NE,KAB)	0 1589
DO 65 J=1,NE	0 1590
JP6 = J + 6	0 1591
65 CMGA2(JP6) = TWC*CM2(J)*DSQRT(CMGA2(JP6))	0 1592
READ (NIT,1002) (UM2(J),J=1,NE)	0 1593
DO 66 J=1,NE	0 1594
JP6 = J + 6	0 1595
66 A(J,J) = UM2(J)*CMGA2(JP6)	0 1596
61 WRITE (NTAPE1) ((A(I,J),J=1,NE),I=1,NE)	0 1597
WRITE (NCT,3019)	0 1598

	CALL WRITES (A ,NE ,NE ,KAB)	0 1599
C		C 1600
	READ (NIT,1002) (UM2(J),J=1,NE)	0 1601
	WRITE (NTAPE1) (OM2(J),J=1,NE)	0 1602
	WRITE (NCT,3020)	0 1603
	CALL WRITES (OM2, 1,NE , 1)	0 1604
	READ (NIT,1002) (CM2(J),J=1,NE)	0 1605
	WRITE (NTAPE1) (OM2(J),J=1,NE)	C 1606
	WRITE (NCT,3021)	0 1607
	CALL WRITES (OM2, 1,NE , 1)	0 1608
C		0 1609
	NHB = NHP01(NB00)	0 1610
	NSB = NSP01(NB00)	0 1611
CCC	NHB IS NO. OF P-Q HINGES ON THE BODY, NOT TO INCLUDE HINGE 1	0 1612
	WRITE (NTAPE1) NHB	0 1613
	DO 110 L=1,NHB	0 1614
		0 1615
	READ (NIT,1001) NOH,ITYPE,JCINT	0 1616
	ISUMRY(L,1) = NOH	0 1617
	ISUMRY(L,2) = ITYPE	0 1618
	ISUMRY(L,3) = JCINT	0 1619
	IF (NOH .LT. 2 .OR. NOH .GT. NH) GO TO 998	0 1620
	LR = 6*(NOH - 2) + 1	0 1621
	LD = 7*(NOH - 2) + 1	0 1622
	IF (ITUPCL(1,NOH) .EQ. NB00) GO TO 112	0 1623
	IF (ITUPCL(2,NOH) .NE. NB00) GO TO 998	0 1624
	LR = LR + 1	0 1625
	LD = LD + 1	0 1626
112	JHX = JOINT	C 1627
	JHY = JHX + NX	0 1628
	JHZ = JHY + NX	0 1629
	JSX = JHZ + NX	0 1630
	JSY = JSX + NX	0 1631
	JSZ = JSY + NX	0 1632
	CH(1,LD) = B(JHY,J)	0 1633
	CH(2,LD) = B(JHZ,J)	0 1634
	CH(3,LD) = B(JHX,J)	0 1635
	READ (NIT,1002) (UM2(J),J=1,3)	C 1636
	ASUMRY(L,1) = UM2(1)	0 1637
	ASUMRY(L,2) = UM2(2)	0 1638
	ASUMRY(L,3) = UM2(3)	0 1639
	CALL ROTTR (3,ITYPE,UM2,RH(1,1,LR),DUM,DJM)	0 1640
	DO 115 J=1,NE	0 1641
	JPC = J + 6	0 1642
	BC(1,J) = B(JHX,JPC)	0 1643
	BC(2,J) = B(JHY,JPC)	C 1644
	BC(3,J) = B(JHZ,JPC)	0 1645
	BC(4,J) = B(JSX,JPC)	0 1646
	BC(5,J) = B(JSY,JPC)	0 1647
115	BC(6,J) = B(JSZ,JPC)	0 1648
	WRITE (NTAPE1) NOH	0 1649
	WRITE (NTAPE1) ((BC(I,J),J=1,NE),I=1,3)	0 1650
	WRITE (NTAPE1) ((BC(I,J),J=1,NE),I=4,6)	0 1651
110	CONTINUE	0 1652
	WRITE (NCT,3022) NB00	0 1653
	CALL WRITES (ISUMRY,NHB,3,KSUMRY)	0 1654
	CALL WRITES (ASUMRY,NHB,J,KSUMRY)	0 1655
C		0 1656
	WRITE (NTAPE1) NSB	0 1657
	IF (NSB .EQ. 0) RETURN	0 1658

CO 120 L=1,NSB	0 1659
READ (NIT,1001) NOS, ITYPE, JOINT	0 1660
ISUMRY(L,1) = NOS	0 1661
ISUMRY(L,2) = ITYPE	0 1662
ISUMRY(L,3) = JOINT	0 1663
IF (IFTSMW(NOS) .NE. NBOD) GO TO 998	0 1664
LR = 2*NOS - 1	0 1665
JHX = JOINT	0 1666
JHY = JHX + NX	0 1667
JHZ = JHY + NX	0 1668
JSX = JHZ + NX	0 1669
JSY = JSX + NX	0 1670
JSZ = JSY + NX	0 1671
CS(1,LR) = B(JHY,3)	0 1672
DS(2,LR) = B(JHZ,1)	0 1673
CS(3,LR) = B(JHX,2)	0 1674
READ (NIT,1002) (UM2(J),J=1,3)	0 1675
ASUMRY(L,1) = UM2(1)	0 1676
ASUMRY(L,2) = UM2(2)	0 1677
ASUMRY(L,3) = UM2(3)	0 1678
CALL ROTTR (3, ITYPE, UM2, RS(1,1,LR), CUM, DJM)	0 1679
GO 125 J=1,NE	0 1680
JP6 = J + 6	0 1681
BC(1,J) = B(JHX,JP6)	0 1682
EG(2,J) = B(JHY,JP6)	0 1683
BC(3,J) = B(JHZ,JP6)	0 1684
BC(4,J) = B(JSX,JP6)	0 1685
BC(5,J) = B(JSY,JP6)	0 1686
125 BC(6,J) = B(JSZ,JP6)	0 1687
WRITE (NTAPE1) NOS	0 1688
WRITE (NTAPE1) ((BC(I,J),J=1,NE),I=1,3)	0 1689
WRITE (NTAPE1) ((BC(I,J),J=1,NE),I=4,6)	0 1690
120 CONTINUE	0 1691
WRITE (NCT,302J) NBOD	0 1692
CALL WRITIS (ISUMRY,NSB,3,KSUMRY)	0 1693
CALL WRITES (ASUMRY,NSB,3,KSUMRY)	0 1694
C	0 1695
RETURN	0 1696
C	0 1697
998 WRITE (NCT,2001)	0 1698
2001 FORMAT (1H1,31HTOPOLOGY ERROR. PROGRAM STOPPED)	0 1699
STCP	0 1700
999 WRITE (NCT,2002)	0 1701
2002 FORMAT (1H1,47HMORE THAN NMUBOD MODES SELECTED,PROGRAM STOPPED)	0 1702
STOP	0 1703
C	0 1704
END	0 1705
SUBROUTINE MSMODL (NBOD)	0 1706
C	0 1707
IMPLICIT REAL*8(A-H,O-Z)	0 1708
C	0 1709
COMMON /NHNS /	0 1710
* NHPCI(6), NSPOI(6)	12 1711
COMMON /NUMBR5/	0 1712
DEBUG # 16	

```

*      ZRC,ONE,TWO,THREE                                0 1713
*      COMMON /SPECIF/                                    0 1714
*      BETAH(6, 6),BLTAND(6, 6),AMO(2, 5),RH(3,3,30),RS(3,3,30), 16 1715
*      LH(3,35),JS(3,30),IML(3, 5),NMCW(6, 6),IFTSMA(15), 17 1716
*      NJ,NH,NSPT,NCHMU,NDELTA,ITCPOL(2, 6),IRGFLX( 6),IHDATA(7, 6), 18 1717
*      LUCC(14),LENU(14),NO,NBETA,NLAM,NEQ 19 1718
*      COMMON /SUMMARY/                                    0 1719
*      ASUMRY(15,6),ISUMRY(15,3),KSUMRY 9E 1720
*      COMMON /TAPE0/                                      0 1721
*      NTAPE1,NTAPE2,NTAPE3                                0 1722
C                                                    0 1723
*      DIMENSION A( 500,12),B( 500,12),C( 500,12),AMU( 18, 18) 64 1724
*      DIMENSION WS( 500,13),OVEC( 500),WV(3) 65 1725
C                                                    0 1726
*      DATA KJCINT, KMODE 0 1727
*      /      500,      12      / 66 1728
*      DATA NIT,NUT /3,6/ 0 1729
C                                                    0 1730
1001 FORMAT(1E15) 0 1731
1002 FORMAT(8E10,3) 0 1732
3022 FORMAT (//5X $HFOR BODY 13,29H THE P-Q HINGE NO., THE EULER 0 1733
* 57H ROTATION TYPE AND THE JOINT NO. CORRESPONDING TO THE P-Q,/ 0 1734
* 5X54H HINGE APPEAR IN THE FOLLOWING INTEGER ARRAY WHICH IS 0 1735
* 49HFOLLOWED BY AN ARRAY CONTAINING EULER ANGLES THAT,/ 0 1736
* 5X44H POSITION THE HINGE TRIAD WRT THE BODY TRIAD) 0 1737
3023 FORMAT (//5X $HFOR BODY 13,32H THE SENSOR POINT NO., THE EULER 0 1738
*60H ROTATION TYPE AND THE JOINT NO. CORRESPONDING TO THE SENSOR,/ 0 1739
*6X53FPPOINT APPEAR IN THE FOLLOWING INTEGER ARRAY WHICH IS 0 1740
* 49HFOLLOWED BY AN ARRAY CONTAINING EULER ANGLES THAT,/ 0 1741
* 5X45H POSITION THE SENSOR TRIAD WRT THE BODY TRIAD) 0 1742
C                                                    0 1743
*      KA = KJOINT 0 1744
*      KB = KJOINT 0 1745
*      KC = KJOINT C 1746
*      KWS = KJOINT 0 1747
*      KAMU = KMODE + 6 0 1748
C                                                    0 1749
C*** INITIALIZE NTAPE2, NTAPE3 0 1750
*      REWIND NTAPE2 0 1751
*      REWIND NTAPE3 0 1752
C                                                    0 1753
C***** 0 1754
C                                                    0 1755
C*** NJ = NUMBER OF MASS POINTS ON BODY I 0 1756
C*** NE = NUMBER OF ELASTIC MODES RETAINED FOR BODY I 0 1757
*      NE = IRGFLX(NBOD) 0 1758
*      NE6 = NE + 6 0 1759
C                                                    0 1760
C*** READ FORM TAPE OR CARDS - CREATE NTAPE2 0 1761
C                                                    0 1762
C      MASSES 0 1763
*      CALL READ (A,N1,N2,KJOINT,KMODE) 0 1764
*      NJ = N1 0 1765
*      DO 2 I=1,NJ 0 1766
*      IF(A(I,1) .LT. ZRO) GO TO 995 0 1767
2 CONTINUE 0 1768
*      WRITE(NTAPE2) (A(I,1),I=1,NJ) 0 1769
C                                                    0 1770
C      INERTIAS 0 1771
*      CALL READ (A,N1,N2,KJOINT,KMODE) 0 1772

```


IF(N1 .NE. NJ .OR. N2 .NE. 6) GO TO 999	0 1773
WRITE(NTAPE2) ((A(1,J),J=1,6),I=1,NJ)	0 1774
C	0 1775
C STATIC MOMENTS - GEOMETRIC COORDINATES	0 1776
DO 5 K=1,2	0 1777
CALL READ (A,N1,N2,KJOINT,KMODE)	0 1778
IF(N1 .NE. NJ .OR. N2 .NE. 3) GO TO 999	0 1779
WRITE(NTAPE2) ((A(1,J),J=1,3),I=1,NJ)	0 1780
5 CONTINUE	0 1781
C	0 1782
C MODAL AMPLITUDES	0 1783
DO 10 K=1,6	0 1784
CALL READ (A,N1,N2,KJOINT,KMODE)	0 1785
IF(N1 .NE. NJ .OR. N2 .NE. NE) GO TO 999	0 1786
WRITE(NTAPE2) ((A(1,J),J=1,NE),I=1,NJ)	0 1787
10 CONTINUE	0 1788
C	0 1789
C STIFFNESS - DAMPING	0 1790
DO 20 K=1,2	0 1791
CALL READ (A,N1,N2,KJOINT,KMODE)	0 1792
IF(N1 .NE. NE .OR. N2 .NE. NE) GO TO 999	0 1793
WRITE(NTAPE2) ((A(1,J),J=1,NE),I=1,NE)	0 1794
20 CONTINUE	0 1795
C	0 1796
DO 47 I=1,NJ	0 1797
47 UVEC(I) = ONE	0 1798
C	0 1799
REWIND NTAPE2	0 1800
NREC2 = 0	0 1801
C	0 1802
C*** CREATE NTAPE3	0 1803
CALL CRET3 (NREC2,NJ,NE,A,B,WS,KA,KB,KWS)	0 1804
C	0 1805
REWIND NTAPE3	0 1806
NREC3 = 0	0 1807
C	0 1808
C*** FORM MUZERO MATRIX	0 1809
CALL ZERO(AMU,NE6,NE6,KAMU)	0 1810
C	0 1811
CALL CREMUJ(NREC3,NJ,UVEC,A,WS,AMU,KA,KWS,KAMU)	0 1812
C	0 1813
C*** FORM A0 AND D0 COEFFICIENTS	0 1814
CALL CREAD0(NREC3,NJ,NE,UVEC,A,B,C,WS,AMU,KA,KC,KWS,KAMU)	0 1815
C	0 1816
C*** FORM E COEFFICIENTS	0 1817
CALL CREE (NREC3,NREC2,NJ,NE,A,B,C,AMU,KA,KB,KC,KAMU)	0 1818
C	0 1819
C*** SYMMETRIZE AMU	0 1820
DO 48 I=1,NE6	0 1821
DO 48 J=1,NE6	0 1822
48 AMU(J,I) = AMU(I,J)	0 1823
C	0 1824
CALL WRITE(AMU,NE6,NE6,3HMUG,KAMU)	0 1825
WRITE(NTAPE1) ((AMU(I,J),J=1,NE6),I=1,NE6)	0 1826
C	0 1827
C*** FORM ACCF	0 1828
CALL CREA (NREC3,NJ,NE,UVEC,A,B,KA,KB,KWS)	0 1829
C	0 1830
C*** FORM BCCF	0 1831
CALL CREB (NREC3,NREC2,NJ,NE,A,B,WS,KA,KB,KWS)	0 1832

C		0 1833
C***	FORM CCOF	0 1834
	CALL CREC (NREC3,NREC2,NJ,NE,A,B,C,AMU,KA,KB,KC,KAMU)	0 1835
C		0 1836
C***	FETCH AND STORE STIFFNESS AND DAMPING	0 1837
	CALL FETCH(NTAPE2,11,NREC2,AMU,NE,NE,KAMU)	0 1838
	WRITE(NTAPE1) ((AMU(1,J),J=1,NE),I=1,NE)	0 1839
	CALL FETCH(NTAPE2,12,NREC2,AMU,NE,NE,KAMU)	0 1840
	WRITE(NTAPE1) ((AMU(1,J),J=1,NE),I=1,NE)	0 1841
C		0 1842
C***	READ AND STORE INITIAL CONDITIONS	0 1843
	READ(NIT,1002) (A(1,J),J=1,NE)	0 1844
	READ(NIT,1002) (A(2,J),J=1,NE)	0 1845
	CALL WRITE(A(1,1),1,NE,4HXED,KA)	0 1846
	CALL WRITE(A(2,1),1,NE,4HXED,KA)	0 1847
	WRITE(NTAPE1) (A(1,J),J=1,NE)	0 1848
	WRITE(NTAPE1) (A(2,J),J=1,NE)	0 1849
C		0 1850
C***	HINGE LOOP *****	0 1851
C		0 1852
	NHB = NRPDI(NBOD)	0 1853
	WRITE(NTAPE1) NHB	0 1854
C		0 1855
	DO 150 L=1,NHB	0 1856
	READ(NIT,1001) NOH,ITYPE,JOINT	0 1857
	ISUMRY(L,1) = NOH	0 1858
	ISUMRY(L,2) = ITYPE	0 1859
	ISUMRY(L,3) = JOINT	0 1860
	IF(NCH.LT.2 .OR. NOH.GT.NH) GO TO 995	0 1861
	LR = 6*(NOH-2) + 1	0 1862
	LD = 7*(NOH-2) + 1	0 1863
	IF(1TOPUL(1,NOH).EQ.NBOD) GO TO 152	0 1864
	IF(1TOPUL(2,NOH).NE.NBOD) GO TO 996	0 1865
	LR = LR+1	0 1866
	LD = LD+1	0 1867
	152 CONTINUE	0 1868
C		0 1869
	CH(1,LD) = WS(JOINT,11)	0 1870
	CH(2,LD) = WS(JOINT,12)	0 1871
	CH(3,LD) = WS(JOINT,13)	0 1872
C		0 1873
C***	READ ANGLES	0 1874
	READ(NIT,1002) (WV(J),J=1,3)	0 1875
	ASUMFY(L,1) = WV(1)	0 1876
	ASUMFY(L,2) = WV(2)	0 1877
	ASUMFY(L,3) = WV(3)	0 1878
C		0 1879
	CALL ROTTR (3,ITYPE,WV,RH(1,1,LR),DUM,DUM)	0 1880
C		0 1881
C***	READ HX,HY,HZ	0 1882
	CALL FETCH(NTAPE2,5,NREC2,A,NJ,NE,KA)	0 1883
	CALL FETCH(NTAPE2,6,NREC2,B,NJ,NE,KB)	0 1884
	CALL FETCH(NTAPE2,7,NREC2,C,NJ,NE,KC)	0 1885
C		0 1886
	DO 154 J=1,NE	0 1887
	AMU(1,J) = A(JOINT,J)	0 1888
	AMU(2,J) = B(JOINT,J)	0 1889
	154 AMU(3,J) = C(JOINT,J)	0 1890
C		0 1891
C***	READ SIGX,SIGY,SIGZ	0 1892

	CALL FETCH(NTAPE2, 8,NREC2,A,NJ,NE,KA)	0 1893
	CALL FETCH(NTAPE2, 9,NREC2,B,NJ,NE,KB)	0 1894
	CALL FETCH(NTAPE2,10,NREC2,C,NJ,NE,KC)	0 1895
C		0 1896
	DO 155 J=1,NE	0 1897
	AMU(4,J) = A(JOINT,J)	0 1898
	AMU(5,J) = B(JOINT,J)	0 1899
155	AMU(6,J) = C(JOINT,J)	0 1900
C		0 1901
	WRITE(NTAPE1) NOH	0 1902
	WRITE(NTAPE1) ((AMU(I,J),J=1,NE),I=1,3)	0 1903
	WRITE(NTAPE1) ((AMU(I,J),J=1,NE),I=4,6)	0 1904
C		0 1905
150	CONTINUE	0 1906
	WRITE (NBT,3022) NBOU	0 1907
	CALL WRITIS (ISUMRY,NHB,3,KSUMRY)	0 1908
	CALL WRITES (ASUMRY,NHB,3,KSUMRY)	0 1909
C		0 1910
C***	SENSOR LOOP *****	0 1911
C		0 1912
	NSE = NSPOI(NBOU)	0 1913
	WRITE(NTAPE1) NSB	0 1914
	IF(NSE .EQ. 0) RETURN	0 1915
C		0 1916
	DO 160 L=1,NSB	0 1917
	READ(NIT,1001) NOS,ITYPE,JOINT	0 1918
	ISUMRY(L,1) = NOS	0 1919
	ISUMRY(L,2) = ITYPE	0 1920
	ISUMRY(L,3) = JOINT	0 1921
	IF(IFTSMW(NOS) .NL. NBOU) GO TO 996	0 1922
	LR = 2*NOS - 1	0 1923
	DS(1,LR) = WS(JOINT,11)	0 1924
	DS(2,LR) = WS(JOINT,12)	0 1925
	DS(3,LR) = WS(JOINT,13)	0 1926
C		0 1927
C***	READ ANGLES	0 1928
	READ(NIT,1002) (WV(J),J=1,3)	0 1929
	ASUMRY(L,1) = WV(1)	0 1930
	ASUMRY(L,2) = WV(2)	0 1931
	ASUMRY(L,3) = WV(3)	0 1932
C		0 1933
	CALL ROTTR (3,ITYPE,WV,RS(1,1,LR),DUM,DUM)	0 1934
C		0 1935
C***	READ FX,HY,HZ	0 1936
	CALL FETCH(NTAPE2, 5,NREC2,A,NJ,NE,KA)	0 1937
	CALL FETCH(NTAPE2, 6,NREC2,B,NJ,NE,KB)	0 1938
	CALL FETCH(NTAPE2, 7,NREC2,C,NJ,NE,KC)	0 1939
C		0 1940
	DO 164 J=1,NE	0 1941
	AMU(1,J) = A(JOINT,J)	0 1942
	AMU(2,J) = B(JOINT,J)	0 1943
164	AMU(3,J) = C(JOINT,J)	0 1944
C		0 1945
C***	READ SIGX,SIGY,SIGZ	0 1946
	CALL FETCH(NTAPE2, 8,NREC2,A,NJ,NE,KA)	0 1947
	CALL FETCH(NTAPE2, 9,NREC2,B,NJ,NE,KB)	0 1948
	CALL FETCH(NTAPE2,10,NREC2,C,NJ,NE,KC)	0 1949
C		0 1950
	DO 165 J=1,NE	0 1951
	AMU(4,J) = A(JOINT,J)	0 1952

AMU(5,J) = U(JOINT,J)	0 1953
165 AMU(6,J) = C(JOINT,J)	0 1954
C	0 1955
WRITE(NTAPE1) NOS	0 1956
WRITE(NTAPE1) ((AMU(1,J),J=1,NE),I=1,3)	0 1957
WRITE(NTAPE1) ((AMU(1,J),J=1,NE),I=4,6)	0 1958
C	0 1959
160 CONTINUE	0 1960
WRITE (NC1,J023) N800	0 1961
CALL WRITIS (ISUMRY,NS0,3,KSUMRY)	0 1962
CALL WRITLS (ASUMRY,NS0,3,KSUMRY)	0 1963
C	0 1964
RETURN	0 1965
C	0 1966
995 WRITE(NOT,2001)	0 1967
2001 FORMAT(1H1,45HNEGATIVE OR ZERO LUMPED MASS, PROGRAM STOPPED)	0 1968
STOP	0 1969
995 WRITE(NOT,2003)	0 1970
2003 FORMAT(1H1,31H1OPOLOGY ERROR, PROGRAM STOPPED)	0 1971
STOP	0 1972
995 WRITE(NOT,2004)	0 1973
2004 FORMAT(1H1,41HERROR IN INPUT TO MSMOUL, PROGRAM STOPPED)	0 1974
STOP	0 1975
C	0 1976
END	0 1977

SUBROUTINE CRET3(NREC2,NJ,NE,A,B,WS,KA,KB,KWS)	0 1978
C	0 1979
C	0 1980
IMPLICIT REAL*8(A-H,O-Z)	0 1981
C	0 1982
LET:	0 1983
C	0 1984
N = JOINT NUMBER N=1,...,NJ	0 1985
M(N) = JOINT MASS	0 1986
C	0 1987
JXX(N) -JXY(N) -JXZ(N)	0 1988
-JXY(N) JYY(N) -JYZ(N) = JOINT INERTIA TENSOR	0 1989
-JXZ(N) -JYZ(N) JZZ(N)	0 1990
C	0 1991
0 SZ(N) -SY(N)	0 1992
-SZ(N) 0 SX(N) = JOINT MASS MOMENT TENSOR	0 1993
SY(N) -SX(N) 0	0 1994
C	0 1995
P1(N) = JOINT MODAL X DISPLACEMENT COMPONENTS	0 1996
P2(N) = JOINT MODAL Y DISPLACEMENT COMPONENTS	0 1997
P3(N) = JOINT MODAL Z DISPLACEMENT COMPONENTS	0 1998
P4(N) = JOINT MODAL X ROTATION COMPONENTS	0 1999
P5(N) = JOINT MODAL Y ROTATION COMPONENTS	0 2000
P6(N) = JOINT MODAL Z ROTATION COMPONENTS	0 2001
X(N) = JOINT X LOCATION	0 2002
Y(N) = JOINT Y LOCATION	0 2003
Z(N) = JOINT Z LOCATION	0 2004
C	0 2005
C	0 2006
THEN:	

```

C          UN NTAPE3 RECORD 1  MATRIX  M * P1          0 2007
C          UN NTAPE3 RECORD 2  MATRIX  SY * P1          0 2008
C          UN NTAPE3 RECORD 3  MATRIX  SZ * P1          0 2009
C          UN NTAPE3 RECORD 4  MATRIX  M * P2          0 2010
C          UN NTAPE3 RECORD 5  MATRIX  SX * P2          0 2011
C          UN NTAPE3 RECORD 6  MATRIX  SZ * P2          0 2012
C          UN NTAPE3 RECORD 7  MATRIX  M * P3          0 2013
C          UN NTAPE3 RECORD 8  MATRIX  SX * P3          0 2014
C          UN NTAPE3 RECORD 9  MATRIX  SY * P3          0 2015
C          UN NTAPE3 RECORD 10 MATRIX  JXX * P4          0 2016
C          UN NTAPE3 RECORD 11 MATRIX  JXY * P4          0 2017
C          UN NTAPE3 RECORD 12 MATRIX  JXZ * P4          0 2018
C          UN NTAPE3 RECORD 13 MATRIX  SY * P4          0 2019
C          UN NTAPE3 RECORD 14 MATRIX  SZ * P4          0 2020
C          UN NTAPE3 RECORD 15 MATRIX  JXY * P5          0 2021
C          UN NTAPE3 RECORD 16 MATRIX  JYY * P5          0 2022
C          UN NTAPE3 RECORD 17 MATRIX  JYZ * P5          0 2023
C          UN NTAPE3 RECORD 18 MATRIX  SX * P5          0 2024
C          UN NTAPE3 RECORD 19 MATRIX  SZ * P5          0 2025
C          UN NTAPE3 RECORD 20 MATRIX  JXZ * P6          0 2026
C          UN NTAPE3 RECORD 21 MATRIX  JYZ * P6          0 2027
C          UN NTAPE3 RECORD 22 MATRIX  JZZ * P6          0 2028
C          UN NTAPE3 RECORD 23 MATRIX  SX * P6          0 2029
C          UN NTAPE3 RECORD 24 MATRIX  SY * P6          0 2030
C          UN NTAPE3 RECORD 25 MATRIX  M * X            0 2031
C          UN NTAPE3 RECORD 26 MATRIX  M * Y            0 2032
C          UN NTAPE3 RECORD 27 MATRIX  M * Z            0 2033
C                                                    0 2034
COMMON /NUMBRS/ ZRO,CNE,TWO,TRES          0 2035
COMMON /TAPE3/  NTAPE1,NTAPE2,NTAPE3      0 2036
C                                                    0 2037
C          DIMENSION A(KA,1),B(KB,1),WS(KWS,1)        0 2038
C          LOGICAL DEBUG,LEQU                0 2039
C          COMMON /LDEBUG/  DEBUG(120)        0 2040
C          EQUIVALENCE (LEQU,DEBUG(17))      0 2041
C          DATA NJ/6/                       0 2042
C          IF(LEQU) WRITE(NOT,1000)           0 2043
1000 FORMAT (' SUBROUTINE CRET3')            0 2044
C                                                    0 2045
C***  FBTCF M                                       0 2046
      CALL FBTCF(NTAPE2, 1,NREC2,WS(1, 1),NJ,1,KWS)  0 2047
C***  FBTCF JXX,.....JYZ                        0 2048
      CALL FBTCF(NTAPE2, 2,NREC2,WS(1, 2),NJ,6,KWS)  0 2049
C***  FBTCF SX,SY,SZ                             0 2050
      CALL FBTCF(NTAPE2, 3,NREC2,WS(1, 8),NJ,3,KWS)  0 2051
C***  FBTCF GEOMETRY                             0 2052
      CALL FBTCF(NTAPE2, 4,NREC2,WS(1,11),NJ,3,KWS)  0 2053
C***  FBTCF HX                                    0 2054
      CALL FBTCF(NTAPE2, 5,NREC2,A,NJ,NE,KA)          0 2055
      CALL STORE(NTAPE3,WS(1, 1),A,B,NJ,NE,KWS,<A,KJ) 0 2056
      IF(LEQU) CALL WRITE(B,NJ,NE,6H M*P1,KB)         0 2057
      CALL STORE(NTAPE3,WS(1, 9),A,B,NJ,NE,KWS,KA,KJ) 0 2058
      IF(LEQU) CALL WRITE(B,NJ,NE,6H SY*P1,KB)        0 2059
      CALL STORE(NTAPE3,WS(1,10),A,B,NJ,NE,KWS,KA,KJ) 0 2060
      IF(LEQU) CALL WRITE(B,NJ,NE,6H SZ*P1,KB)        0 2061
C***  FBTCF HY                                    0 2062
      CALL FBTCF(NTAPE2, 6,NREC2,A,NJ,NE,KA)          0 2063
      CALL STORE(NTAPE3,WS(1, 1),A,B,NJ,NE,KWS,KA,KJ) 0 2064
      IF(LEQU) CALL WRITE(B,NJ,NE,6H M*P2,KB)         0 2065
      CALL STORE(NTAPE3,WS(1, 8),A,B,NJ,NE,KWS,KA,KJ) 0 2066

```

IF (LEQU) CALL WRITE(B,NJ,NE,6H SX*P2,KB)	0 2067
CALL STORE(NTAPE3,WS(1,10),A,B,NJ,NE,KWS,KA,K3)	0 2068
IF (LEQU) CALL WRITE(B,NJ,NE,6H SZ*P2,KB)	0 2069
C*** FBTCF HZ	0 2070
CALL FETCH(NTAPE2, 7,NREC2,A,NJ,NE,KA)	0 2071
CALL STORE(NTAPE3,WS(1, 1),A,B,NJ,NE,KWS,KA,K3)	0 2072
IF (LEQU) CALL WRITE(B,NJ,NE,6H M*P3,KB)	0 2073
CALL STORE(NTAPE3,WS(1, 8),A,B,NJ,NE,KWS,KA,K3)	0 2074
IF (LEQU) CALL WRITE(B,NJ,NE,6H SX*P3,KB)	0 2075
CALL STORE(NTAPE3,WS(1, 9),A,B,NJ,NE,KWS,KA,K3)	0 2076
IF (LEQU) CALL WRITE(B,NJ,NE,6H SY*P3,KB)	0 2077
C*** FBTCF SIGX	0 2078
CALL FETCH(NTAPE2, 8,NREC2,A,NJ,NE,KA)	0 2079
CALL STORE(NTAPE3,WS(1, 2),A,B,NJ,NE,KWS,KA,K3)	0 2080
IF (LEQU) CALL WRITE(B,NJ,NE,6HJXX*P4,KB)	0 2081
CALL STORE(NTAPE3,WS(1, 5),A,B,NJ,NE,KWS,KA,K3)	0 2082
IF (LEQU) CALL WRITE(B,NJ,NE,6HJXY*P4,KB)	0 2083
CALL STORE(NTAPE3,WS(1, 6),A,B,NJ,NE,KWS,KA,K3)	0 2084
IF (LEQU) CALL WRITE(B,NJ,NE,6HJXZ*P4,KB)	0 2085
CALL STORE(NTAPE3,WS(1, 9),A,B,NJ,NE,KWS,KA,K3)	0 2086
IF (LEQU) CALL WRITE(B,NJ,NE,6H SY*P4,KB)	0 2087
CALL STORE(NTAPE3,WS(1,10),A,B,NJ,NE,KWS,KA,K3)	0 2088
IF (LEQU) CALL WRITE(B,NJ,NE,6H SZ*P4,KB)	0 2089
C*** FBTCF SIGY	0 2090
CALL FETCH(NTAPE2, 9,NREC2,A,NJ,NE,KA)	0 2091
CALL STORE(NTAPE3,WS(1, 5),A,B,NJ,NE,KWS,KA,K3)	0 2092
IF (LEQU) CALL WRITE(B,NJ,NE,6HJXY*P5,KB)	0 2093
CALL STORE(NTAPE3,WS(1, 3),A,B,NJ,NE,KWS,KA,K3)	0 2094
IF (LEQU) CALL WRITE(B,NJ,NE,6HJYY*P5,KB)	0 2095
CALL STORE(NTAPE3,WS(1, 7),A,B,NJ,NE,KWS,KA,K3)	0 2096
IF (LEQU) CALL WRITE(B,NJ,NE,6HJYZ*P5,KB)	0 2097
CALL STORE(NTAPE3,WS(1, 8),A,B,NJ,NE,KWS,KA,K3)	0 2098
IF (LEQU) CALL WRITE(B,NJ,NE,6H SX*P5,KB)	0 2099
CALL STORE(NTAPE3,WS(1,10),A,B,NJ,NE,KWS,KA,K3)	0 2100
IF (LEQU) CALL WRITE(B,NJ,NE,6H SZ*P5,KB)	0 2101
C*** FBTCF SIGZ	0 2102
CALL FETCH(NTAPE2,10,NREC2,A,NJ,NE,KA)	0 2103
CALL STORE(NTAPE3,WS(1, 6),A,B,NJ,NE,KWS,KA,K3)	0 2104
IF (LEQU) CALL WRITE(B,NJ,NE,6HJXZ*P6,KB)	0 2105
CALL STORE(NTAPE3,WS(1, 7),A,B,NJ,NE,KWS,KA,K3)	0 2106
IF (LEQU) CALL WRITE(B,NJ,NE,6HJYZ*P6,KB)	0 2107
CALL STORE(NTAPE3,WS(1, 4),A,B,NJ,NE,KWS,KA,K3)	0 2108
IF (LEQU) CALL WRITE(B,NJ,NE,6HJZZ*P6,KB)	0 2109
CALL STORE(NTAPE3,WS(1, 8),A,B,NJ,NE,KWS,KA,K3)	0 2110
IF (LEQU) CALL WRITE(B,NJ,NE,6H SX*P6,KB)	0 2111
CALL STORE(NTAPE3,WS(1, 9),A,B,NJ,NE,KWS,KA,K3)	0 2112
IF (LEQU) CALL WRITE(B,NJ,NE,6H SY*P6,KB)	0 2113
CALL STORE(NTAPE3,WS(1, 1),WS(1,11),B,NJ,1,KWS,KWS,KB)	0 2114
IF (LEQU) CALL WRITE(B,NJ,1,6H M*X,KB)	0 2115
CALL STORE(NTAPE3,WS(1, 1),WS(1,12),B,NJ,1,KWS,KWS,KB)	0 2116
IF (LEQU) CALL WRITE(B,NJ,1,6H M*Y,KB)	0 2117
CALL STORE(NTAPE3,WS(1, 1),WS(1,13),B,NJ,1,KWS,KWS,KB)	0 2118
IF (LEQU) CALL WRITE(B,NJ,1,6H M*Z,KB)	0 2119
C	0 2120
RETURN	0 2121
END	0 2122

```

SUBROUTINE CRLMU0(NREC3,NJ,UVEC,A,WS,AMU,KA,KWS,KAMU)      0 2123
C                                                           0 2124
C                                                           DEBUG # 18      0 2125
IMPLICIT REAL*8(A-H,O-Z)      0 2126
C                                                           0 2127
C      USE THE LUMPED PARAMETER DATA ON NTAPE3, ALONG WITH EQUATIONS 0 2128
C      11-30,31,35 AND 37 TO COMPUTE THE INERTIA TENSOR, STATIC MASS 0 2129
C      MOMENT AND TOTAL MASS FOR THE UNDEFORMED FLEXIBLE BODY.      0 2130
C      THESE TERMS ARE STORED AS A 6 BY 6 MATRIX AND DEFINE THE      0 2131
C      UPPER LEFT 2 BY 2 PARTITION OF EQUATION 11-37      0 2132
C                                                           0 2133
C                                                           0 2134
COMMON /NUMBRS/ ZRG,ONE,TWO,TRES      0 2135
COMMON /TAPEND/ NTAPE1,NTAPE2,NTAPE3      0 2136
C                                                           0 2137
DIMENSION A(KA,1),WS(KWS,1),AMU(KAMU,1),UVEC(1)      0 2138
C                                                           0 2139
C***  FETCH M*RHGX,M*RHGY,M*RHGZ      0 2140
CALL FETCH(NTAPE3,25,NREC3,A(1,1),NJ,1,KA)      0 2141
CALL FETCH(NTAPE3,26,NREC3,A(1,2),NJ,1,KA)      0 2142
CALL FLTCH(NTAPE3,27,NREC3,A(1,3),NJ,1,KA)      0 2143
C***  FORM JXXC      0 2144
CALL SATB( ONE,WS(1,13), A(1, 3),AMU(1,1),NJ, 1, 1,KWS, KA,KAMU)      0 2145
CALL SATB( ONE,WS(1,12), A(1, 2),AMU(1,1),NJ, 1, 1,KWS, KA,KAMU)      0 2146
CALL SATB( TWO,WS(1,13),WS(1,10),AMU(1,1),NJ, 1, 1,KWS,KWS,KAMU)      0 2147
CALL SATB( TWO,WS(1,12),WS(1, 9),AMU(1,1),NJ, 1, 1,KWS,KWS,KAMU)      0 2148
CALL SATB( ONE,UVEC      ,WS(1, 2),AMU(1,1),NJ, 1, 1,KWS,KWS,KAMU)      0 2149
C***  FORM JXYC      0 2150
CALL SATB(-ONE,WS(1,12), A(1, 1),AMU(1,2),NJ, 1, 1,KWS, KA,KAMU)      0 2151
CALL SATB(-ONE,WS(1,12),WS(1, 8),AMU(1,2),NJ, 1, 1,KWS,KWS,KAMU)      0 2152
CALL SATB(-ONE,WS(1,11),WS(1, 9),AMU(1,2),NJ, 1, 1,KWS,KWS,KAMU)      0 2153
CALL SATB(-ONE,UVEC      ,WS(1, 5),AMU(1,2),NJ, 1, 1,KWS,KWS,KAMU)      0 2154
C***  FORM JXZC      0 2155
CALL SATB(-ONE,WS(1,13), A(1, 1),AMU(1,3),NJ, 1, 1,KWS, KA,KAMU)      0 2156
CALL SATB(-ONE,WS(1,13),WS(1, 8),AMU(1,3),NJ, 1, 1,KWS,KWS,KAMU)      0 2157
CALL SATB(-ONE,WS(1,11),WS(1,10),AMU(1,3),NJ, 1, 1,KWS,KWS,KAMU)      0 2158
CALL SATB(-ONE,UVEC      ,WS(1, 6),AMU(1,3),NJ, 1, 1,KWS,KWS,KAMU)      0 2159
C***  FORM JYYC      0 2160
CALL SATB( ONE,WS(1,13), A(1, 3),AMU(2,2),NJ, 1, 1,KWS, KA,KAMU)      0 2161
CALL SATB( ONE,WS(1,11), A(1, 1),AMU(2,2),NJ, 1, 1,KWS, KA,KAMU)      0 2162
CALL SATB( TWO,WS(1,13),WS(1,10),AMU(2,2),NJ, 1, 1,KWS,KWS,KAMU)      0 2163
CALL SATB( TWO,WS(1,11),WS(1, 8),AMU(2,2),NJ, 1, 1,KWS,KWS,KAMU)      0 2164
CALL SATB( ONE,UVEC      ,WS(1, 3),AMU(2,2),NJ, 1, 1,KWS,KWS,KAMU)      0 2165
C***  FORM JYZC      0 2166
CALL SATB(-ONE,WS(1,13), A(1, 2),AMU(2,3),NJ, 1, 1,KWS, KA,KAMU)      0 2167
CALL SATB(-ONE,WS(1,13),WS(1, 9),AMU(2,3),NJ, 1, 1,KWS,KWS,KAMU)      0 2168
CALL SATB(-ONE,WS(1,12),WS(1,10),AMU(2,3),NJ, 1, 1,KWS,KWS,KAMU)      0 2169
CALL SATB(-ONE,UVEC      ,WS(1, 7),AMU(2,3),NJ, 1, 1,KWS,KWS,KAMU)      0 2170
C***  FORM JZZC      0 2171
CALL SATB( ONE,WS(1,12), A(1, 2),AMU(3,3),NJ, 1, 1,KWS, KA,KAMU)      0 2172
CALL SATB( ONE,WS(1,11), A(1, 1),AMU(3,3),NJ, 1, 1,KWS, KA,KAMU)      0 2173
CALL SATB( TWO,WS(1,12),WS(1, 9),AMU(3,3),NJ, 1, 1,KWS,KWS,KAMU)      0 2174
CALL SATB( TWO,WS(1,11),WS(1, 8),AMU(3,3),NJ, 1, 1,KWS,KWS,KAMU)      0 2175
CALL SATB( ONE,UVEC      ,WS(1, 4),AMU(3,3),NJ, 1, 1,KWS,KWS,KAMU)      0 2176
C***  FORM SXO      0 2177
CALL SATB( ONE,UVEC      ,WS(1, 8),AMU(3,5),NJ, 1, 1,KWS,KWS,KAMU)      0 2178
CALL SATB( ONE,WS(1,11),WS(1, 1),AMU(3,5),NJ, 1, 1,KWS,KWS,KAMU)      0 2179
C***  FORM SYO      0 2180
CALL SATB( ONE,UVEC      ,WS(1, 9),AMU(1,6),NJ, 1, 1,KWS,KWS,KAMU)      0 2181
CALL SATB( ONE,WS(1,12),WS(1, 1),AMU(1,6),NJ, 1, 1,KWS,KWS,KAMU)      0 2182

```

```

C*** FORM S20
CALL SATB( ONE,UVEC      ,WS(1,13),AMU(2,4),NJ, 1, 1,KWS,KWS,KAMU) 0 2183
CALL SATB( ONE,WS(1,13),WS(1, 1),AMU(2,4),NJ, 1, 1,KWS,KWS,KAMU) 0 2184
AMU(2,6) = -AMU(3,5) 0 2186
AMU(3,4) = -AMU(1,5) 0 2187
AMU(1,5) = -AMU(2,4) 0 2188
C*** FORM MASS 0 2189
CALL SATB( ONE,UVEC      ,WS(1, 1),AMU(4,4),NJ, 1, 1,KWS,KWS,KAMU) 0 2190
AMU(5,5) = AMU(4,4) 0 2191
AMU(6,6) = AMU(4,4) 0 2192
C 0 2193
CALL WRITE(AMU(1,1),3,3,SHINER0,KAMU) 0 2194
CALL WRITE(AMU(1,4),3,3,SHSTAT0,KAMU) 0 2195
CALL WRITE(AMU(4,4),3,3,SHMASS0,KAMU) 0 2196
C 0 2197
RETURN 0 2198
END 0 2199

SUBROUTINE CREADJ(NREC3,NJ,NE,UVEC,A,B,C,WS,A4U,KA,KB,KC,KWS,KAMU) 0 2200
C 0 2201
C          DEBUG # 19 0 2202
C 0 2203
C      USE THE LUMPED PARAMETER DATA ON NTAPES, ALONG WITH EQUATIONS 0 2204
C      11-33,36 TO COMPUTE THE 3 BY 1RGFLX(N) MATRIX OF D-COEFFICIENT 0 2205
C      AND THE 3 BY 1RGFLX(N) MATRIX OF A-COEFFICIENTS IN EQ 11-87 0 2206
C      THESE ARE INERTIAL COUPLING COEFFICIENTS WHICH ARE INDEPENDENT 0 2207
C      OF DEFORMATION 0 2208
C 0 2209
C      IMPLICIT REAL*8(A-H,O-Z) 0 2210
C 0 2211
COMMON /NUMBRS/ ZRO,ONE,TWO,TRES 0 2212
COMMON /TAPENO/ NTAPE1,NTAPE2,NTAPE3 0 2213
C 0 2214
C      DIMENSION A(KA,1),B(KB,1),C(KC,1),WS(KWS,1),AMU(KAMU,1),UVEC(1) 0 2215
C 0 2216
C*** FETCH M*FX,M*HY,M*HZ 0 2217
CALL FETCH(NTAPE3, 1,NREC3,A,NJ,NE,KA) 0 2218
CALL FETCH(NTAPE3, 4,NREC3,B,NJ,NE,KB) 0 2219
CALL FETCH(NTAPE3, 7,NREC3,C,NJ,NE,KC) 0 2220
CALL SATB( ONE,UVEC      ,A,AMU(4,7),NJ, 1,NE,KWS,KA,KAMU) 0 2221
CALL SATB( ONE,UVEC      ,B,AMU(5,7),NJ, 1,NE,KWS,KB,KAMU) 0 2222
CALL SATB( ONE,UVEC      ,C,AMU(6,7),NJ, 1,NE,KWS,KC,KAMU) 0 2223
CALL SATB( ONE,WS(1,13),A,AMU(2,7),NJ, 1,NE,KWS,KA,KAMU) 0 2224
CALL SATB(-ONE,WS(1,12),A,AMU(3,7),NJ, 1,NE,KWS,KA,KAMU) 0 2225
CALL SATB(-ONE,WS(1,13),B,AMU(1,7),NJ, 1,NE,KWS,KB,KAMU) 0 2226
CALL SATB( ONE,WS(1,11),B,AMU(3,7),NJ, 1,NE,KWS,KB,KAMU) 0 2227
CALL SATB( ONE,WS(1,12),C,AMU(1,7),NJ, 1,NE,KWS,KC,KAMU) 0 2228
CALL SATB(-ONE,WS(1,11),C,AMU(2,7),NJ, 1,NE,KWS,KC,KAMU) 0 2229
C*** FETCH SY*SIGX,SZ*SIGX 0 2230
CALL FETCH(NTAPE3,13,NREC3,A,NJ,NE,KA) 0 2231
CALL FETCH(NTAPE3,14,NREC3,B,NJ,NE,KB) 0 2232
CALL SATB( ONE,UVEC      ,A,AMU(6,7),NJ, 1,NE,KWS,KA,KAMU) 0 2233
CALL SATB(-ONE,UVEC      ,B,AMU(5,7),NJ, 1,NE,KWS,KB,KAMU) 0 2234
CALL SATB( ONE,WS(1,12),A,AMU(1,7),NJ, 1,NE,KWS,KA,KAMU) 0 2235
CALL SATB( ONE,WS(1,13),B,AMU(1,7),NJ, 1,NE,KWS,KB,KAMU) 0 2236

```


	CALL SATE(-ONE,WS(1,11),A,AMU(2,7),NJ, 1,NE,KWS,KA,KAMU)	0 2237
	CALL SATB(-ONE,WS(1,11),B,AMU(3,7),NJ, 1,NE,KWS,KB,KAMU)	0 2238
C***	FETCH SX*SIGY,SZ*SIGY	0 2239
	CALL FETCH(NTAPE3,18,NREC3,A,NJ,NE,KA)	0 2240
	CALL FETCH(NTAPE3,19,NREC3,B,NJ,NE,KB)	0 2241
	CALL SATE(-ONE,UVEC ,A,AMU(6,7),NJ, 1,NE,KWS,KA,KAMU)	0 2242
	CALL SATE(ONE,UVEC ,B,AMU(4,7),NJ, 1,NE,KWS,KB,KAMU)	0 2243
	CALL SATE(-ONE,WS(1,12),A,AMU(1,7),NJ, 1,NE,KWS,KA,KAMU)	0 2244
	CALL SATE(ONE,WS(1,13),B,AMU(2,7),NJ, 1,NE,KWS,KB,KAMU)	0 2245
	CALL SATE(-ONE,WS(1,11),A,AMU(2,7),NJ, 1,NE,KWS,KA,KAMU)	0 2246
	CALL SATB(-ONE,WS(1,12),B,AMU(3,7),NJ, 1,NE,KWS,KB,KAMU)	0 2247
C***	FETCH SX*SIGZ,SY*SIGZ	0 2248
	CALL FETCH(NTAPE3,23,NREC3,A,NJ,NE,KA)	0 2249
	CALL FETCH(NTAPE3,24,NREC3,B,NJ,NE,KB)	0 2250
	CALL SATB(ONE,UVEC ,A,AMU(5,7),NJ, 1,NE,KWS,KA,KAMU)	0 2251
	CALL SATE(-ONE,UVEC ,B,AMU(4,7),NJ, 1,NE,KWS,KB,KAMU)	0 2252
	CALL SATE(-ONE,WS(1,13),A,AMU(1,7),NJ, 1,NE,KWS,KA,KAMU)	0 2253
	CALL SATE(-ONE,WS(1,13),B,AMU(2,7),NJ, 1,NE,KWS,KB,KAMU)	0 2254
	CALL SATE(ONE,WS(1,11),A,AMU(3,7),NJ, 1,NE,KWS,KA,KAMU)	0 2255
	CALL SATE(ONE,WS(1,12),B,AMU(3,7),NJ, 1,NE,KWS,KB,KAMU)	0 2256
C***	FETCH SY*HX,SZ*HX	0 2257
	CALL FETCH(NTAPE3, 2,NREC3,A,NJ,NE,KA)	0 2258
	CALL FETCH(NTAPE3, 3,NREC3,B,NJ,NE,KB)	0 2259
	CALL SATE(ONE,UVEC ,B,AMU(2,7),NJ, 1,NE,KWS,KB,KAMU)	0 2260
	CALL SATB(-ONE,UVEC ,A,AMU(3,7),NJ, 1,NE,KWS,KA,KAMU)	0 2261
C***	FETCH SX*HY,SZ*HY	0 2262
	CALL FETCH(NTAPE3, 5,NREC3,A,NJ,NE,KA)	0 2263
	CALL FETCH(NTAPE3, 6,NREC3,B,NJ,NE,KB)	0 2264
	CALL SATE(-ONE,UVEC ,B,AMU(1,7),NJ, 1,NE,KWS,KB,KAMU)	0 2265
	CALL SATB(ONE,UVEC ,A,AMU(3,7),NJ, 1,NE,KWS,KA,KAMU)	0 2266
C***	FETCH SX*HZ,SY*HZ	0 2267
	CALL FETCH(NTAPE3, 8,NREC3,A,NJ,NE,KA)	0 2268
	CALL FETCH(NTAPE3, 9,NREC3,B,NJ,NE,KB)	0 2269
	CALL SATE(ONE,UVEC ,B,AMU(1,7),NJ, 1,NE,KWS,KB,KAMU)	0 2270
	CALL SATE(-ONE,UVEC ,A,AMU(2,7),NJ, 1,NE,KWS,KA,KAMU)	0 2271
C***	FETCH JXX*SIGX,JXY*SIGX,JXZ*SIGX	0 2272
	CALL FETCH(NTAPE3,10,NREC3,A,NJ,NE,KA)	0 2273
	CALL FETCH(NTAPE3,11,NREC3,B,NJ,NE,KB)	0 2274
	CALL FETCH(NTAPE3,12,NREC3,C,NJ,NE,KC)	0 2275
	CALL SATE(ONE,UVEC ,A,AMU(1,7),NJ, 1,NE,KWS,KA,KAMU)	0 2276
	CALL SATE(-ONE,UVEC ,B,AMU(2,7),NJ, 1,NE,KWS,KB,KAMU)	0 2277
	CALL SATL(-ONE,UVEC ,C,AMU(3,7),NJ, 1,NE,KWS,KC,KAMU)	0 2278
C***	FETCH JXY*SIGY,JYY*SIGY,JYZ*SIGY	0 2279
	CALL FETCH(NTAPE3,15,NREC3,A,NJ,NE,KA)	0 2280
	CALL FETCH(NTAPE3,16,NREC3,B,NJ,NE,KB)	0 2281
	CALL FETCH(NTAPE3,17,NREC3,C,NJ,NE,KC)	0 2282
	CALL SATE(-ONE,UVEC ,A,AMU(1,7),NJ, 1,NE,KWS,KA,KAMU)	0 2283
	CALL SATE(ONE,UVEC ,B,AMU(2,7),NJ, 1,NE,KWS,KB,KAMU)	0 2284
	CALL SATB(-ONE,UVEC ,C,AMU(3,7),NJ, 1,NE,KWS,KC,KAMU)	0 2285
C***	FETCH JXZ*SIGZ,JYZ*SIGZ,JZZ*SIGZ	0 2286
	CALL FETCH(NTAPE3,20,NREC3,A,NJ,NE,KA)	0 2287
	CALL FETCH(NTAPE3,21,NREC3,B,NJ,NE,KB)	0 2288
	CALL FETCH(NTAPE3,22,NREC3,C,NJ,NE,KC)	0 2289
	CALL SATB(-ONE,UVEC ,A,AMU(1,7),NJ, 1,NE,KWS,KA,KAMU)	0 2290
	CALL SATB(-ONE,UVEC ,B,AMU(2,7),NJ, 1,NE,KWS,KB,KAMU)	0 2291
	CALL SATB(ONE,UVEC ,C,AMU(3,7),NJ, 1,NE,KWS,KC,KAMU)	0 2292
C		0 2293
	CALL WRITE(AMU(1,7),3,NE,SHDUCDEF,KAMU)	0 2294
	CALL WRITE(AMU(4,7),3,NE,SHAUCDEF,KAMU)	0 2295
C		0 2296
	RETURN	0 2297
	END	0 2298

C	SUBROUTINE CREB(NREC3,NREC2,NJ,NE,A,B,C,AMJ,KA,KB,KC,KAMU)	0 2299
C		0 2300
C	DEBUG # 20	0 2301
C	IMPLICIT REAL*8(A-H,U-Z)	0 2302
C		0 2303
C	USE THE LUMPED PARAMETER DATA CN NTAPE3; AND EQ 11-34 TO	0 2304
C	COMPUTE THE IRGFLX(N) BY IRGFLX(N) MODAL MASS MATRIX. THIS	0 2305
C	IS THE E MATRIX IN EQ 11-87	0 2306
C		0 2307
C	COMMON /NUMBRS/ ZRD,UNE,TWG,TRES	0 2308
C	COMMON /TAPENO/ NTAPE1,NTAPE2,NTAPE3	0 2309
C		0 2310
C	DIMENSION A(KA,1),B(KB,1),C(KC,1),AMU(KAMU,1)	0 2311
C		0 2312
C***	FETCH M*HX,SZ*SIGY,SY*SIGZ	0 2313
	CALL FETCH(NTAPE3,1,NREC3,A,NJ,NE,KA)	0 2314
	CALL FETCH(NTAPE3,19,NREC3,B,NJ,NE,KB)	0 2315
	CALL FETCH(NTAPE3,24,NREC3,C,NJ,NE,KC)	0 2316
	CALL ADD3(UNE,A,UNE,B,-UNE,C,NJ,NE,KA)	0 2317
C***	FETCH HX	0 2318
	CALL FETCH(NTAPE2,5,NREC2,B,NJ,NE,KB)	0 2319
	CALL SATB(UNE,B,A,AMU(7,7),NJ,NE,NE,KB,KA,KAMU)	0 2320
C***	FETCH M*HY,SX*SIGZ,SZ*SIGX	0 2321
	CALL FETCH(NTAPE3,4,NREC3,A,NJ,NE,KA)	0 2322
	CALL FETCH(NTAPE3,23,NREC3,B,NJ,NE,KB)	0 2323
	CALL FETCH(NTAPE3,14,NREC3,C,NJ,NE,KC)	0 2324
	CALL ADD3(UNE,A,UNE,B,-UNE,C,NJ,NE,KA)	0 2325
C***	FETCH HY	0 2326
	CALL FETCH(NTAPE2,6,NREC2,B,NJ,NE,KB)	0 2327
	CALL SATB(UNE,B,A,AMU(7,7),NJ,NE,NE,KB,KA,KAMU)	0 2328
C***	FETCH M*HZ,SY*SIGX,SX*SIGY	0 2329
	CALL FETCH(NTAPE3,7,NREC3,A,NJ,NE,KA)	0 2330
	CALL FETCH(NTAPE3,13,NREC3,B,NJ,NE,KB)	0 2331
	CALL FETCH(NTAPE3,18,NREC3,C,NJ,NE,KC)	0 2332
	CALL ADD3(UNE,A,UNE,B,-UNE,C,NJ,NE,KA)	0 2333
C***	FETCH HZ	0 2334
	CALL FETCH(NTAPE2,7,NREC2,B,NJ,NE,KB)	0 2335
	CALL SATB(UNE,B,A,AMU(7,7),NJ,NE,NE,KB,KA,KAMU)	0 2336
C***	FETCH JXX*SIGX,JXY*SIGY,JXZ*SIGZ	0 2337
	CALL FETCH(NTAPE3,10,NREC3,A,NJ,NE,KA)	0 2338
	CALL FETCH(NTAPE3,15,NREC3,B,NJ,NE,KB)	0 2339
	CALL FETCH(NTAPE3,20,NREC3,C,NJ,NE,KC)	0 2340
	CALL ADD3(UNE,A,-UNE,B,-UNE,C,NJ,NE,KA)	0 2341
C***	FETCH SY*HZ,SZ*HY	0 2342
	CALL FETCH(NTAPE3,9,NREC3,B,NJ,NE,KB)	0 2343
	CALL FETCH(NTAPE3,6,NREC3,C,NJ,NE,KC)	0 2344
	CALL ADD3(UNE,A,UNE,B,-UNE,C,NJ,NE,KA)	0 2345
C***	FETCH SIGX	0 2346
	CALL FETCH(NTAPE2,8,NREC2,B,NJ,NE,KB)	0 2347
	CALL SATB(UNE,B,A,AMU(7,7),NJ,NE,NE,KB,KA,KAMU)	0 2348
C***	FETCH JXY*SIGX,JYY*SIGY,JYZ*SIGZ	0 2349
	CALL FETCH(NTAPE3,11,NREC3,A,NJ,NE,KA)	0 2350

	CALL FETCH(NTAPE3,16,NREC3,B,NJ,NE,KB)	0 2351
	CALL FETCH(NTAPE3,21,NREC3,C,NJ,NE,KC)	0 2352
	CALL ADD3(-ONE,A, ONE,B,-ONE,C,NJ,NE,KA)	0 2353
C***	FETCH SZ*HX, SX*HZ	0 2354
	CALL FETCH(NTAPE3, 3,NREC3,B,NJ,NE,KB)	0 2355
	CALL FETCH(NTAPE3, 8,NREC3,C,NJ,NE,KC)	0 2356
	CALL ADD3(ONE,A, ONE,B,-ONE,C,NJ,NE,KA)	0 2357
C***	FETCH SIGY	0 2358
	CALL FETCH(NTAPE2, 9,NREC2,B,NJ,NE,KB)	0 2359
	CALL SATB(ONE,B,A,AMU(7,7),NJ,NE,NE,KB,KA,KAMU)	0 2360
C***	FETCH JXZ*SIGX, JYZ*SIGY, JZZ*SIGZ	0 2361
	CALL FETCH(NTAPE3,12,NREC3,A,NJ,NE,KA)	0 2362
	CALL FETCH(NTAPE3,17,NREC3,B,NJ,NE,KB)	0 2363
	CALL FETCH(NTAPE3,22,NREC3,C,NJ,NE,KC)	0 2364
	CALL ADD3(-ONE,A,-ONE,B, ONE,C,NJ,NE,KA)	0 2365
C***	FETCH SX*HY, SY*HX	0 2366
	CALL FETCH(NTAPE3, 5,NREC3,B,NJ,NE,KB)	0 2367
	CALL FETCH(NTAPE3, 2,NREC3,C,NJ,NE,KC)	0 2368
	CALL ADD3(ONE,A,ONE,B,-ONE,C,NJ,NE,KA)	0 2369
C***	FETCH SIGZ	0 2370
	CALL FETCH(NTAPE2,10,NREC2,D,NJ,NE,KB)	0 2371
	CALL SATB(ONE,B,A,AMU(7,7),NJ,NE,NE,KB,KA,KAMU)	0 2372
	CALL WRITE(AMU(7,7),NE,NE,SHECCCF,KAMU)	0 2373
C		0 2374
	RETURN	0 2375
	END	0 2376
	SUBROUTINE CREA(NREC3,NJ,NE,UEVC,A,B,KA,KB,KWS)	0 2377
C		0 2378
C	DEBUG # 21	0 2379
	IMPLICIT REAL*8(A-H,O-Z)	0 2380
C		0 2381
C	USE THE LUMPED PARAMETER DATA ON NTAPES, AND EQ 11-33,35	0 2382
C	ALONG WITH A-12 TO A-20 TO COMPUTE THE ALPHA COEFFICIENTS OF	0 2383
C	EQ 11-88. USED TO DEFINE THE LINEAR DEPENDENCE OF BODY'S	0 2384
C	STATIC MASS MOMENT ON DEFORMATION	0 2385
C		0 2386
	COMMON /NUMBRS/ ZRC,ONE,TWO,TRES	0 2387
	COMMON /TAPEND/ NTAPE1,NTAPE2,NTAPE3	0 2388
C		0 2389
	DIMENSION A(KA,1),B(KB,1),UEVC(1)	0 2390
C		0 2391
	CALL ZCRC(A,9,NE,KA)	0 2392
C***	FETCH M*HX	0 2393
	CALL FETCH(NTAPE3, 1,NREC3,B,NJ,NE,KB)	0 2394
	CALL SATB(-ONE,UEVC,B,A(6,1),NJ,1,NE,KWS,KB,KA)	0 2395
C***	FETCH M*HY	0 2396
	CALL FETCH(NTAPE3, 4,NREC3,B,NJ,NE,KB)	0 2397
	CALL SATB(ONE,UEVC,B,A(3,1),NJ,1,NE,KWS,KB,KA)	0 2398
C***	FETCH M*HZ	0 2399
	CALL FETCH(NTAPE3, 7,NREC3,B,NJ,NE,KB)	0 2400
	CALL SATB(-ONE,UEVC,B,A(2,1),NJ,1,NE,KWS,KB,KA)	0 2401
	DO 45 J=1,NE	0 2402
	A(4,J) = -A(2,J)	0 2403
	A(7,J) = -A(3,J)	0 2404

45	A(2,J) = -A(6,J)	C 2405
C		0 2406
	CALL WRITE(A,9,NE,4HACCF,KA)	C 2407
	WRITE(NTAPE1) ((A(1,J),J=1,NE),I=1,9)	C 2408
C		0 2409
	RETURN	0 2410
	END	0 2411
	SUBROUTINE CREB(NREC3,NREC2,NJ,NE,A,H,WS,KA,KB,KWS)	C 2412
C		0 2413
C	DEBUG # 22	0 2414
	IMPLICIT REAL*8(A-H,O-Z)	0 2415
C		0 2416
C	USE THE LUMPED PARAMETER DATA ON NTAPES AND EQS 11-31,32,37	C 2417
C	ALONG WITH A-21 TO A-26 TO COMPUTE THE B COEFFICIENTS OF	0 2418
C	EQ 11-38. USED TO DEFINE THE LINEAR DEPENDENCE OF BODY'S	0 2419
C	INERTIA TENSOR ON DEFORMATION	0 2420
C		0 2421
	COMMON /NUMBRS/ ZRO,ONE,TWO,TRES	0 2422
	COMMON /TAPENO/ NTAPE1,NTAPE2,NTAPE3	0 2423
C		0 2424
	DIMENSION A(KA,1),B(KB,1),WS(KWS,1)	0 2425
C		0 2426
	CALL ZERO(U,6,NE,KE)	0 2427
C***	FETCH M*RHDX,M*RHLY,M*RHZ	0 2428
	CALL FETCH(NTAPE3,25,NREC3,A(1,1),NJ,1,KA)	0 2429
	CALL FETCH(NTAPE3,26,NREC3,A(1,2),NJ,1,KA)	0 2430
	CALL FETCH(NTAPE3,27,NREC3,A(1,3),NJ,1,KA)	0 2431
	DO 52 I=1,NJ	0 2432
	WS(I, 8) = WS(I, 8) + A(1,1)	0 2433
	WS(I, 9) = WS(I, 9) + A(1,2)	0 2434
52	WS(I,10) = WS(I,10) + A(1,3)	0 2435
C***	FETCH HX	0 2436
	CALL FETCH(NTAPE2, 5,NREC2,A,NJ,NE,KA)	0 2437
	CALL SATE(ONE,WS(1, 8),A,B(2,1),NJ,1,NE,KWS,KA,KB)	0 2438
	CALL SATE(ONE,WS(1, 8),A,B(3,1),NJ,1,NE,KWS,KA,KB)	0 2439
	CALL SATE(ONE,WS(1, 9),A,B(4,1),NJ,1,NE,KWS,KA,KB)	0 2440
	CALL SATE(ONE,WS(1,10),A,B(5,1),NJ,1,NE,KWS,KA,KB)	0 2441
C***	FETCH HY	0 2442
	CALL FETCH(NTAPE2, 6,NREC2,A,NJ,NE,KA)	0 2443
	CALL SATE(ONE,WS(1, 9),A,B(1,1),NJ,1,NE,KWS,KA,KB)	0 2444
	CALL SATE(ONE,WS(1, 9),A,B(3,1),NJ,1,NE,KWS,KA,KB)	0 2445
	CALL SATE(ONE,WS(1, 8),A,B(4,1),NJ,1,NE,KWS,KA,KB)	0 2446
	CALL SATE(ONE,WS(1,10),A,B(6,1),NJ,1,NE,KWS,KA,KB)	0 2447
C***	FETCH HZ	0 2448
	CALL FETCH(NTAPE2, 7,NREC2,A,NJ,NE,KA)	0 2449
	CALL SATE(ONE,WS(1,10),A,B(1,1),NJ,1,NE,KWS,KA,KB)	0 2450
	CALL SATE(ONE,WS(1,10),A,B(2,1),NJ,1,NE,KWS,KA,KB)	0 2451
	CALL SATE(ONE,WS(1, 8),A,B(5,1),NJ,1,NE,KWS,KA,KB)	0 2452
	CALL SATE(ONE,WS(1, 9),A,B(6,1),NJ,1,NE,KWS,KA,KB)	0 2453
C		0 2454
	CALL WRITE(B,6,NE,4HBCUF,KB)	0 2455
	WRITE(NTAPE1) ((B(1,J),J=1,NE),I=1,6)	0 2456
C		0 2457
	RETURN	0 2458
	END	0 2459

SUBROUTINE CREC(NREC3,NREC2,NJ,NE,A,B,C,AMU,KA,KB,KC,KAMU)	0 2460
C	0 2461
C	0 2462
IMPLICIT REAL*8(A-H,O-Z)	0 2463
C	0 2464
C	0 2465
C	0 2466
C	0 2467
C	0 2468
C	0 2469
C	0 2470
C	0 2471
C	0 2472
COMMON /NUMBRS/ ZRO,ONE,TWO,TRES	0 2473
COMMON /TAPENG/ NTAPE1,NTAPE2,NTAPE3	0 2474
C	0 2475
DIMENSION A(KA,1),B(KB,1),C(KC,1),AMU(KAMU,1)	0 2476
C	0 2477
C*** FORM CCCF1 = CXY	0 2478
CALL ZERG(AMU,NE,NE,KAMU)	0 2479
C*** FETCH M*FY,SX*SIGZ,SZ*SIGX	0 2480
CALL FETCH(NTAPE3, 4,NREC3,A,NJ,NE,KA)	0 2481
CALL FETCH(NTAPE3,23,NREC3,B,NJ,NE,KB)	0 2482
CALL FETCH(NTAPE3,14,NREC3,C,NJ,NE,KC)	0 2483
CALL ADD3(ONE,A, ONE,B,-ONE,C,NJ,NE,KA)	0 2484
C*** FETCH HX	0 2485
CALL FETCH(NTAPE2, 5,NREC2,B,NJ,NE,KB)	0 2486
CALL SATB(ONE,B,A,AMU,NJ,NE,KB,KA,KAMU)	0 2487
C*** FETCH M*HX,SZ*SIGY,SY*SIGZ	0 2488
CALL FETCH(NTAPE3, 1,NREC3,A,NJ,NE,KA)	0 2489
CALL FETCH(NTAPE3,19,NREC3,B,NJ,NE,KB)	0 2490
CALL FETCH(NTAPE3,24,NREC3,C,NJ,NE,KC)	0 2491
CALL ADD3(ONE,A, ONE,B,-ONE,C,NJ,NE,KA)	0 2492
C*** FETCH HY	0 2493
CALL FETCH(NTAPE2, 6,NREC2,B,NJ,NE,KB)	0 2494
CALL SATB(-ONE,B,A,AMU,NJ,NE,KB,KA,KAMU)	0 2495
C	0 2496
CALL WRITE(AMU,NE,NE,3HCXY,KAMU)	0 2497
WRITE(NTAPE1) ((AMU(I,J),J=1,NE),I=1,NE)	0 2498
C	0 2499
C*** FORM CCCF2 = CXZ	0 2500
CALL ZERG(AMU,NE,NE,KAMU)	0 2501
C*** FETCH HZ	0 2502
CALL FETCH(NTAPE2, 7,NREC2,B,NJ,NE,KB)	0 2503
CALL SATB(ONE,B,A,AMU,NJ,NE,KB,KA,KAMU)	0 2504
C*** FETCH M*HZ,SY*SIGX,SX*SIGY	0 2505
CALL FETCH(NTAPE3, 7,NREC3,A,NJ,NE,KA)	0 2506
CALL FETCH(NTAPE3,15,NREC3,B,NJ,NE,KB)	0 2507
CALL FETCH(NTAPE3,18,NREC3,C,NJ,NE,KC)	0 2508
CALL ADD3(ONE,A, ONE,B,-ONE,C,NJ,NE,KA)	0 2509
C*** FETCH HX	0 2510
CALL FETCH(NTAPE2, 5,NREC2,B,NJ,NE,KB)	0 2511
CALL SATB(-ONE,B,A,AMU,NJ,NE,KB,KA,KAMU)	0 2512

C	CALL WRITE(AMU,NE,NE,3HCXZ,KAMU)	0 2513
	WRITE(NTAPE1) ((AMU(I,J),J=1,NE),I=1,NE)	0 2514
C		0 2515
C***	FORM CCOF3 = CYZ	0 2516
	CALL ZERO(AMU,NE,NE,KAMU)	0 2517
C***	FETCH HY	0 2518
	CALL FETCH(NTAPE2, 6,NREC2,B,NJ,NE,KB)	0 2519
	CALL SATB(ONE,B,A,AMU,NJ,NE,NE,KB,KA,KAMU)	0 2520
C***	FETCH M*HY,SX*SIGZ,SZ*SIGX	0 2521
	CALL FETCH(NTAPE3, 4,NREC3,A,NJ,NE,KA)	0 2522
	CALL FETCH(NTAPE3,23,NREC3,B,NJ,NE,KB)	0 2523
	CALL FETCH(NTAPE3,14,NREC3,C,NJ,NE,KC)	0 2524
	CALL ADD3(ONE,A, ONE,B,-ONE,C,NJ,NE,KA)	0 2525
C***	FETCH HZ	0 2526
	CALL FETCH(NTAPE2, 7,NREC2,B,NJ,NE,KB)	0 2527
	CALL SATB(-ONE,B,A,AMU,NJ,NE,NE,KB,KA,KAMU)	0 2528
C		0 2529
	CALL WRITE(AMU,NE,NE,3HCYZ,KAMU)	0 2530
	WRITE(NTAPE1) ((AMU(I,J),J=1,NE),I=1,NE)	0 2531
C		0 2532
C***	FORM CCOF4 = C11	0 2533
	CALL ZERO(AMU,NE,NE,KAMU)	0 2534
C***	FETCH M*HY AND HY	0 2535
	CALL FETCH(NTAPE3, 4,NREC3,A,NJ,NE,KA)	0 2536
	CALL FETCH(NTAPE2, 6,NREC2,B,NJ,NE,KB)	0 2537
	CALL SATB(ONE,B,A,AMU,NJ,NE,NE,KB,KA,KAMU)	0 2538
C***	FETCH M*FZ AND HZ	0 2539
	CALL FETCH(NTAPE3, 7,NREC3,A,NJ,NE,KA)	0 2540
	CALL FLTCH(NTAPE2, 7,NREC2,B,NJ,NE,KB)	0 2541
	CALL SATB(ONE,B,A,AMU,NJ,NE,NE,KB,KA,KAMU)	0 2542
C		0 2543
	CALL WRITE(AMU,NE,NE,3HC11,KAMU)	0 2544
	WRITE(NTAPE1) ((AMU(I,J),J=1,NE),I=1,NE)	0 2545
C		0 2546
C***	FORM CCOF5 = C22	0 2547
	CALL ZERO(AMU,NE,NE,KAMU)	0 2548
	CALL SATB(ONE,B,A,AMU,NJ,NE,NE,KB,KA,KAMU)	0 2549
C***	FETCH M*FX AND HX	0 2550
	CALL FETCH(NTAPE3, 1,NREC3,A,NJ,NE,KA)	0 2551
	CALL FLTCH(NTAPE2, 5,NREC2,B,NJ,NE,KB)	0 2552
	CALL SATB(ONE,B,A,AMU,NJ,NE,NE,KB,KA,KAMU)	0 2553
C		0 2554
	CALL WRITE(AMU,NE,NE,3HC22,KAMU)	0 2555
	WRITE(NTAPE1) ((AMU(I,J),J=1,NE),I=1,NE)	0 2556
C		0 2557
C***	FORM CCOF6 = C33	0 2558
	CALL ZERO(AMU,NE,NE,KAMU)	0 2559
	CALL SATB(ONE,B,A,AMU,NJ,NE,NE,KB,KA,KAMU)	0 2560
C***	FETCH M*HY AND HY	0 2561
	CALL FETCH(NTAPE3, 4,NREC3,A,NJ,NE,KA)	0 2562
	CALL FLTCH(NTAPE2, 6,NREC2,B,NJ,NE,KB)	0 2563
	CALL SATB(ONE,B,A,AMU,NJ,NE,NE,KB,KA,KAMU)	0 2564
C		0 2565
	CALL WRITE(AMU,NE,NE,3HC33,KAMU)	0 2566
	WRITE(NTAPE1) ((AMU(I,J),J=1,NE),I=1,NE)	0 2567
C		0 2568
C***	FORM CCOF7 = C12	0 2569
	CALL ZERO(AMU,NE,NE,KAMU)	0 2570
C***	FETCH M*FX	0 2571
		0 2572

	CALL FETCH(NTAPE3, 1,NREC3,A,NJ,NE,KA)	0 2573
	CALL SATE(ONE,B,A,AMU,NJ,NE,NE,KB,KA,KAMU)	0 2574
C		0 2575
	CALL WRITE(AMU,NE,NE,3HC12,KAMU)	0 2576
	WRITE(NTAPE1) ((AMU(I,J),J=1,NE),I=1,NE)	0 2577
C		0 2578
C***	FORM CCGF8 = C13	0 2579
	CALL ZERC(AMU,NE,NE,KAMU)	0 2580
C***	FETCH HZ	0 2581
	CALL FETCH(NTAPE2, 7,NREC2,B,NJ,NE,KB)	0 2582
	CALL SATB(ONE,B,A,AMU,NJ,NE,NE,KB,KA,KAMU)	0 2583
C		0 2584
	CALL WRITE(AMU,NE,NE,3HC13,KAMU)	0 2585
	WRITE(NTAPE1) ((AMU(I,J),J=1,NE),I=1,NE)	0 2586
C		0 2587
C***	FORM CCGF9 = C23	0 2588
	CALL ZERC(AMU,NE,NE,KAMU)	0 2589
C***	FETCH M*HY	0 2590
	CALL FETCH(NTAPE3, 4,NREC3,A,NJ,NE,KA)	0 2591
	CALL SATB(ONE,B,A,AMU,NJ,NE,NE,KB,KA,KAMU)	0 2592
C		0 2593
	CALL WRITE(AMU,NE,NE,3HC23,KAMU)	0 2594
	WRITE(NTAPE1) ((AMU(I,J),J=1,NE),I=1,NE)	0 2595
C		0 2596
	RETURN	0 2597
	END	0 2598

	SUBROUTINE STORE(NTAPE,A,B,Z,NA,NCB,KRA,KR3,KRZ)	0 2599
C		0 2600
C		0 2601
	IMPLICIT REAL*8(A-H,O-Z)	0 2602
C		0 2603
C	MATRIX PRODUCT Z = A*B WITH A = DIAGONAL AND STORED AS VECTOR	0 2604
C	PRODUCT WRITTEN BY ROWS ON TAPE	0 2605
C	WHERE A = INPUT (NA,NA)	0 2606
C	B = INPUT (NA,NCB)	0 2607
C	Z = OUTPUT (NA,NCB)	0 2608
C		0 2609
	DIMENSION A(KRA,1),B(KRB,1),Z(KRZ,1)	0 2610
C		0 2611
	DO 10 I=1,NA	0 2612
	S = A(I,1)	0 2613
	DO 10 J=1,NCB	0 2614
10	Z(I,J) = S*B(I,J)	0 2615
C		0 2616
	WRITE(NTAPE) ((Z(I,J),J=1,NCB),I=1,NA)	0 2617
C		0 2618
	RETURN	0 2619
	END	0 2620

C	SUBROUTINE FETCH(NTAPE,NREC,NRECO,A,NRA,NCA,KRA)	C 2621
	DEBUG # 25	0 2622
C	IMPLICIT REAL*8(A-H,O-Z)	0 2623
C		0 2624
C	FETCH MATRIX FROM NTAPE - ASSUMED WRITTEN BY ROWS	0 2625
C	WHERE NREC = INPUT RECORD NUMBER DESIRED	0 2626
C	NRECO = LAST OUTPUT RECORD NUMBER FETCHED FROM NTAPE	0 2627
C	USED TO AVOID EXCESSIVE TAPE SEARCH	0 2628
C	A = ARRAY WHERE RECORD STORED (NRA,NCA)	0 2629
C		0 2630
C	DIMENSION A(KRA,1)	0 2631
C		0 2632
C	IF(NREC - NRECO) 1,2,3	0 2633
C		0 2634
C	1 REWIND NTAPE	0 2635
	NSKIP = NREC - 1	0 2636
	IF(NSKIP .EQ. 0) GO TO 20	0 2637
	DO 10 K=1,NSKIP	0 2638
10	READ(NTAPE) DUM	0 2639
	GO TO 20	0 2640
C		0 2641
C	2 BACKSPACE NTAPE	0 2642
	GO TO 20	0 2643
C		0 2644
C	3 NSKIP = NREC - NRECO - 1	0 2645
	IF(NSKIP .EQ. 0) GO TO 20	0 2646
	DO 11 K=1,NSKIP	0 2647
11	READ(NTAPE) DUM	0 2648
	GO TO 20	0 2649
C		0 2650
20	NRECO = NREC	0 2651
C		0 2652
C	READ(NTAPE) ((A(I,J),J=1,NCA),I=1,NRA)	0 2653
C		0 2654
C	RETURN	0 2655
	END	0 2656

C	SUBROUTINE SATB(S,A,B,Z,NRA,NCA,NCB,KRA,KRB,KRZ)	0 2657
C		0 2658
C	DEBUG # 26	0 2659
C	IMPLICIT REAL*8(A-H,O-Z)	0 2660
C		0 2661
C	MATRIX PRODUCT Z(NEW) = Z(OLD) + S*A *B	0 2662
C	WHERE S = SCALAR	0 2663
C	A = INPUT (NRA,NCA)	0 2664
C	B = INPUT (NRA,NCB)	0 2665
C	Z = OUTPUT (NCA,NCB)	0 2666
C		0 2667
C	DIMENSION A(KRA,1),B(KRB,1),Z(KRZ,1)	0 2668
C		0 2669
	DO 20 I=1,NCA	0 2670
	DO 20 J=1,NCB	0 2671
	DO 20 K=1,NRA	0 2672
20	Z(I,J) = Z(I,J) + S*A(K,I)*B(K,J)	0 2673
C		0 2674
C	RETURN	0 2675
	END	0 2676

SUBROUTINE ADD3(ALPHA,A,BETA,B,GAMMA,C,NR,NC,KR)	0 2677
IMPLICIT REAL*8(A-H,O-Z)	0 2678
C	0 2679
C	0 2680
C	0 2681
C	0 2682
C	0 2683
C	0 2684
C	0 2685
C	0 2686
C	0 2687
C	0 2688
10 A(I,J) = ALPHA*A(I,J) + BETA*B(I,J) + GAMMA*C(I,J)	0 2689
C	0 2690
C	0 2691
C	0 2692
END	
SUBROUTINE MRIGID (N)	0 2693
C	0 2694
C	0 2695
C	0 2696
C	0 2697
C	0 2698
C	0 2699
C	12 2700
C	0 2701
C	16 2702
C	17 2703
C	18 2704
C	19 2705
C	0 2706
C	98 2707
C	0 2708
C	0 2709
C	0 2710
C	0 2711
C	0 2712
C	0 2713
C	0 2714
C	0 2715
C	0 2716
C	0 2717
C	0 2718
C	0 2719
C	0 2720
C	0 2721
C	0 2722

```

* 56 ROTATION TYPE APPEAR IN THE FOLLOWING INTEGER ARRAY WHICH / 0 2723
* 5X5 THIS FOLLOWED BY AN ARRAY CONTAINING EULER ANGLES(1,2,3), 0 2724
* 56 HAND POSITION VECTOR COMPONENTS (4,5,6) THAT POSITION THE / 0 2725
* 5) SENSE SENSOR TRIAD WRT THE BODY TRIAD) 0 2726

C 0 2727
C NHB = NUMBER OF HINGES ON BODY N 0 2728
C NSB = NUMBER OF SENSOR POINTS ON BODY N 0 2729
C NHE = NHFOI(N) 0 2730
C NSB = NSFOI(N) 0 2731
C 0 2732
C CREATE 6X6 INERTIA MATRIX 0 2733
C CALL ZERC (AIDER,6,6,6) 0 2734
C 0 2735
C***** READ BODY MASS AND CG LOCATION 0 2736
C CALL READ (V,N1,N2,1,6) 0 2737
C COMPUTE RIGID BODY MASS MOMENT ABOUT CG 0 2738
C FOR GENERALITY INERTIA TENSOR DEFINED ABOUT REFERENCE POINT 0 2739
C DO 5 J=2,4 0 2740
5 V(J) = -V(1)*V(J) 0 2741
CALL SKEWV3 (V(2),AIDER(1,4),1,6) 0 2742
DO 6 I=4,6 0 2743
6 AIDER(1,I) = V(1) 0 2744
C 0 2745
C***** READ INERTIA TENSOR OF BODY RELATIVE TO REFERENCE 0 2746
C POINT. IF NOT CG MASS MOMENT NON-ZERO 0 2747
C CALL READ (V,N1,N2,1,6) 0 2748
C DO 7 I=1,3 0 2749
7 AIDER(1,I) = V(1) 0 2750
AIDER(1,2) = -V(4) 0 2751
AIDER(1,3) = -V(5) 0 2752
AIDER(2,3) = -V(6) 0 2753
C DO 8 I=1,6 0 2754
DO 8 J=1,6 0 2755
8 AIDER(J,I) = AIDER(I,J) 0 2756
WRITE (NTAPE1) ((AIDER(I,J),J=1,6),I=1,6) 0 2757
CALL PAGEHD 0 2758
WRITE (NGT,3001) N 0 2759
C PRINT BODY INERTIA MATRIX FOR BODY N 0 2760
C CALL WRITES (AIDER,6,6,6) 0 2761
C 0 2762
C CYCLE THROUGH HINGES ON BODY N 0 2763
C DO 10 I=1,NHB 0 2764
C 0 2765
C***** READ HINGE NUMBER AND EULER TYPE 0 2766
C READ (NIT,1001) NOH, ITYPE 0 2767
C NOH = HINGE NUMBER 0 2768
C ITYPE = TYPE OF EULER ANGLE SEQUENCE 0 2769
C HINGE TO BODY TRIAD 0 2770
C ISUMRY(1,1) = NOH 0 2771
C ISUMRY(1,2) = ITYPE 0 2772
C IF (NOH .EQ. 1) GO TO 999 0 2773
C LR = 6*(NOH - 2) + 3 0 2774
C LD = 7*(NOH - 2) + 3 0 2775
C IF (ITYPE(1,NOH) .EQ. N) GO TO 12 0 2776
C IF (ITYPE(2,NOH) .NE. N) GO TO 999 0 2777
C LR = LR + 1 0 2778
C LD = LD + 1 0 2779
C 0 2780
C***** READ ANGLES TO ORIENT HINGE AND BODY TRIADS 0 2781
C USE TO COMPUTE TRANSFORMATION MATRIX IN RH 0 2782

```

12 READ (NIT,1002) (V(J),J=1,3)	0 2783
C	0 2784
C***** READ VECTOR POSITION BODY TO HINGE TRIAD	0 2785
C STORE IN DH ARRAY	0 2786
READ (NIT,1002) (DH(J,LD),J=1,3)	0 2787
C	0 2788
DO 11 J=1,3	0 2789
J1 = J + 3	0 2790
ASUMRY(1,J) = V(J)	0 2791
11 ASUMRY(1,J1) = DH(J,LD)	0 2792
C	0 2793
C BRANCH TO ROTTR TO COMPUTE TRANSFORMATION MATRIX	0 2794
C HINGE FRAME TO BODY FRAME	0 2795
C RH = ARRAY TO STORE HINGE TO BODY	0 2796
C TRANSFORMATION MATRICES	0 2797
CALL ROTTR (3,ITYPE,V,RH(1,1,LR),DUM,DUM)	0 2798
C	0 2799
10 CONTINUE	0 2800
C PRINT INITIAL HINGE TO BODY FRAME ORIENTATION DATA	0 2801
WRITE (NOT,3002) N	0 2802
CALL WRITIS (ISUMRY,NHB,2,KSUMRY)	0 2803
CALL WRITES (ASUMRY,NHB,6,KSUMRY)	0 2804
C	0 2805
IF (NSB .EQ. 0) RETURN	0 2806
DO 20 I=1,NSB	0 2807
C NOS = SENSOR POINT NUMBER	0 2808
C	0 2809
READ (NIT,1001) NOS,ITYPE	0 2810
ISUMRY(1,1) = NOS	0 2811
ISUMRY(1,2) = ITYPE	0 2812
IF (IFSMW(NOS) .NE. N) GO TO 999	0 2813
LR = 2*NOS	0 2814
C	0 2815
C***** READ ANGLES TO ORIENT SENSOR AND BODY TRIADS	0 2816
C USE TO COMPUTE TRANSFORMATION MATRIX IN RS	0 2817
READ (NIT,1002) (V(J),J=1,3)	0 2818
C	0 2819
C***** READ VECTOR SENSOR POINT TO BODY TRIAD	0 2820
C STORE IN DS ARRAY	0 2821
READ (NIT,1002) (DS(J,LR),J=1,3)	0 2822
C	0 2823
DO 21 J=1,3	0 2824
J1 = J + 3	0 2825
ASUMRY(1,J) = V(J)	0 2826
21 ASUMRY(1,J1) = DS(J,LR)	0 2827
C	0 2828
C BRANCH TO ROTTR TO GET SENSOR POINT TO BODY FRAME	0 2829
C TRANSFORMATION MATRIX	0 2830
C RS = ARRAY TO STORE SENSOR POINT TO BODY	0 2831
C TRANSFORMATION MATRIX	0 2832
CALL ROTTR (3,ITYPE,V,RS(1,1,LR),DUM,DUM)	0 2833
C	0 2834
20 CONTINUE	0 2835
WRITE (NOT,3003) N	0 2836
CALL WRITIS (ISUMRY,NSB,2,KSUMRY)	0 2837
CALL WRITES (ASUMRY,NSB,6,KSUMRY)	0 2838
RETURN	0 2839
C	0 2840
999 WRITE (NOT,2001)	0 2841
2001 FORMAT (1H1,49HTUPLUGGY ERROR,SUBROUTINE MRIGID, PROGRAM STOPPED)	0 2842
STCP	0 2843
C	0 2844
END	0 2845

C	SUBROUTINE ZERO (Z,NR,NC,KR)	0 2846
		0 2847
	IMPLICIT REAL*8(A-H,O-Z)	0 2848
	DIMENSION Z(KR,1)	0 2849
C		0 2850
C	GENERATE A MATRIX OF ZEROES.	0 2851
C	CODED BY RL WUHLIN. FEB 1965.	0 2852
C		0 2853
C	SUBROUTINE ARGUMENTS	0 2854
C	Z = OUTPUT MATRIX GENERATED. SIZE(NR,NC).	0 2855
C	NR = INPUT NUMBER OF ROWS IN MATRIX Z.	0 2856
C	NC = INPUT NUMBER OF COLS IN MATRIX Z.	0 2857
C	KR = INPUT ROW DIMENSION OF MATRIX Z IN CALLING PROGRAM.	0 2858
C		0 2859
	DO 10 I=1,NR	0 2860
	DO 10 J=1,NC	0 2861
	10 Z(I,J) = 0.0 0	0 2862
	RETURN	0 2863
	END	0 2864

	SUBROUTINE REVISE (A,IVEC,JVEC,Z,NRA,NCA,NRZ,NCZ,KRA,KRZ)	0 2865
		0 2866
C		0 2867
	IMPLICIT REAL*8(A-H,O-Z)	0 2868
	DIMENSION A(KRA,1), IVEC(1), JVEC(1), Z(KRZ,1)	0 2869
	DATA NOT / 0 /	0 2870
C		0 2871
C	REARRANGE ROWS AND COLUMNS OF MATRIX A TO FORM MATRIX Z.	0 2872
C	CALLS PERMA SUBROUTINE ZZDUMB.	0 2873
C	CODED BY RF HRUCA. FEBRUARY 1965.	0 2874
C	LAST REVISION BY RL WUHLIN. OCTOBER 1972.	0 2875
C		0 2876
C	SUBROUTINE ARGUMENTS	0 2877
C	A = INPUT MATRIX TO BE REARRANGED. SIZE(NRA,NCA).	0 2878
C	IVEC = INPUT VECTOR. SIZE(NRA).	0 2879
C	IVEC(1)=ROW POSITION OF A(ROW 1) IN Z.	0 2880
C	IF IVEC(1) IS PLUS, Z(ROW IVEC(1)) = +A(ROW 1).	0 2881
C	IF IVEC(1) IS MINUS, Z(ROW IVEC(1)) = -A(ROW 1).	0 2882
C	IF IVEC(1) IS ZERO, A(ROW 1) IS OMITTED IN Z.	0 2883
C	JVEC = INPUT VECTOR. SIZE(NCA).	0 2884
C	JVEC(J)=COL POSITION OF A(COL J) IN Z.	0 2885
C	IF JVEC(J) IS PLUS, Z(COL JVEC(J)) = +A(COL J).	0 2886
C	IF JVEC(J) IS MINUS, Z(COL JVEC(J)) = -A(COL J).	0 2887
C	IF JVEC(J) IS ZERO, A(COL J) IS OMITTED IN Z.	0 2888
C	Z = OUTPUT RESULT MATRIX. SIZE(NRZ,NCZ).	0 2889
C	NRA = INPUT NUMBER OF ROWS IN MATRIX A.	0 2890

C	NCA	= INPUT	NUMBER OF COLS IN MATRIX A.	0	2891
C	NRZ	= INPUT	NUMBER OF ROWS IN MATRIX Z.	0	2892
C	NCZ	= INPUT	NUMBER OF COLS IN MATRIX Z.	0	2893
C	KRA	= INPUT	ROW DIMENSION OF A IN CALLING PROGRAM.	0	2894
C	KRZ	= INPUT	ROW DIMENSION OF Z IN CALLING PROGRAM.	0	2895
C				0	2896
	DO 10 I=1, NRZ			0	2897
	DO 10 J=1, NCZ			0	2898
	10 Z(I,J) = 0.0 0			0	2899
C				0	2900
	DO 30 IA=1, NRA			0	2901
	IZ = IABS(IVC(IA))			0	2902
	IF (IZ .EQ. 0) GO TO 30			0	2903
C				0	2904
	IF (IZ .GT. NRZ) GO TO 999			0	2905
	DO 25 JA=1, NCA			0	2906
	JZ = IABS(JVEC(JA))			0	2907
	IF (JZ .EQ. 0) GO TO 25			0	2908
C				0	2909
	IF (JZ .GT. NCZ) GO TO 999			0	2910
	SIGN = +1.0 0			0	2911
	IF (IVC(IA).LT.0 .AND. JVEC(JA).GT.0 .OR.			0	2912
	* IVEC(IA).GT.0 .AND. JVEC(JA).LT.0) SIGN=-1.0 0			0	2913
	Z(IZ,JZ) = SIGN*A(IA,JA)			0	2914
	25 CONTINUE			0	2915
	30 CONTINUE			0	2916
	RETURN			0	2917
C				0	2918
	999 WRITE (NOT,1001)			0	2919
	1001 FORMAT (1H1,J2HERROR IN REVISE, PROGRAM STOPPED)			0	2920
	STOP			0	2921
	END			0	2922

	SUBROUTINE GAUSSI (A,R,N,KAR)			0	2923
C			DEBUG # 31	0	2924
C	FIND INVERSE OF MATRIX A RETURN A**(-1) IN R ARRAY			0	2925
C	SAVE PIVOT ELEMENTS IN DRVEC THESE WILL BE USED TO			0	2926
C	COMPUTE DETERMINANT OF A			0	2927
	IMPLICIT REAL*8(A-H,O-Z)			0	2928
	DIMENSION A(KAR,1), R(KAR,1), IVEC(150)			69	2929
	COMMON /DRATIO/			0	2930
	* IFL1,IFL2,DRVEC(150)			100	2931
C				0	2932
	LOGICAL DEBUG,LEQU			0	2933
	COMMON /LDEBUG/ DEBUG(120)			0	2934
	EQUIVALENCE (LEQU,DEBUG(31))			0	2935
	DATA EPS,NOT / 1.0-15, 6/			0	2936
	1001 FORMAT (EX,30HMATRIX SINGULAR IN GAUSSI AT , 15)			0	2937
C				0	2938
	DO 5 I=1,N			0	2939
	DO 7 J=1,N			0	2940
	7 R(I,J) = 0.0 0			0	2941
	5 R(I,I) = 1.0 0			0	2942
	IF (.NOT.LEQU) GO TO 1			0	2943
	WRITE(NOT,1002)			0	2944

CALL WRITES(A,N,N,KAR)	0 2945
1002 FORMAT (' SUBROUTINE GAUSSI, INPUT MATRIX A IS')	0 2946
1 CONTINUE	0 2947
C	0 2948
DO 10 L=1,N	0 2949
JBIG = 1	0 2950
A1 = DABS(A(L,1))	0 2951
DO 15 J=2,N	0 2952
A2 = DABS(A(L,J))	0 2953
IF (A2 .LT. A1) GO TO 15	0 2954
A1 = A2	0 2955
JBIG = J	0 2956
15 CONTINUE	0 2957
IVEC(L) = JBIG	0 2958
IF (A1 .GT. EPS) GO TO 20	0 2959
IF (IFL1 .EQ. 0) GO TO 75	0 2960
C IFL1 = 1 CALL IS FROM NUMS, ZERO PIVOT ELEMENT IMPLIES	0 2961
C ZERO GAIN FOR NUMERATOR	0 2962
IFL2 = 1	0 2963
GO TO 100	0 2964
75 WRITE (NCT,1001) L	0 2965
STOP	0 2966
20 CONTINUE	0 2967
ALJBIG = A(L,JBIG)	0 2968
C STORE ALL PIVOT ELEMENTS IN DRVEC	0 2969
DRVEC(L) = ALJBIG	0 2970
DO 17 J=1,N	0 2971
A(L,J) = A(L,J)/ALJBIG	0 2972
17 R(L,J) = R(L,J)/ALJBIG	0 2973
DO 25 I=1,N	0 2974
AIJBIG = A(I,JBIG)	0 2975
IF (I .EQ. L) GO TO 25	0 2976
DO 30 J=1,N	0 2977
A(I,J) = A(I,J) - AIJBIG*A(L,J)	0 2978
30 R(I,J) = R(I,J) - AIJBIG*R(L,J)	0 2979
25 CONTINUE	0 2980
10 CONTINUE	0 2981
C	0 2982
DO 40 I=1,N	0 2983
IR = IVEC(I)	0 2984
DO 40 J=1,N	0 2985
40 A(IR, J) = R(I,J)	0 2986
DO 50 I=1,N	0 2987
DO 50 J=1,N	0 2988
50 R(I,J) = A(I,J)	0 2989
C	0 2990
SN = 1.00	0 2991
DO 110 L=1,N	0 2992
DO 120 J=1,N	0 2993
IF (IVLC(J) .EQ. L) GO TO 110	0 2994
IF (IVEC(J) .GT. L) SN = -SN	0 2995
120 CONTINUE	0 2996
110 CONTINUE	0 2997
DRVEC(1) = SN*DRVEC(1)	0 2998
100 IFL1 = 0	0 2999
IF (.NOT. LEQU) RETURN	0 3000
WRITE(NCT,1003)	0 3001
1003 FORMAT (' INVERSE OF INPUT MATRIX A IS')	0 3002
CALL WRITES(R,N,N,KAR)	0 3003
WRITE(NCT,1004)	0 3004

1004	FORMAT (' PIVOT ELEMENTS USED TO GET A**(-1) ARE ')	C 3005
	CALL WRITES(DRVEC,1,N,1)	0 3006
	RETURN	0 3007
	END	0 3008
	SUBROUTINE ROTTR (I23, ITYPE, ANG, ROT, RP2, RP3)	0 3009
C	IMPLICIT REAL*8(A-H, O-Z)	0 3010
	DIMENSION ANG(1), ROT(3,3), RP2(3,3), RP3(3,3)	C 3011
	DIMENSION IP(36), A(3,3,4), IV(3)	0 3012
	COMMON /NUMBRS/	0 3013
	* ZRO, ONE, TWO, TRES	0 3014
		0 3015
C		0 3016
C	SUBROUTINE TO COMPUTE 12 TYPES OF EULER ANGLE TRANSFORMATIONS	0 3017
C	AND/OR PI INVERSE TRANSFORMATIONS.	0 3018
C	ALSO IF I23 .EQ. 1, COMPUTE RP2 AND RP3 WHICH MULTIPLY	0 3019
C	D/DT(THETA(2)) AND D/DT(THETA(3)) TO FORM D/DT(PI INVERSE).	C 3020
C	I23 = (1 IF ROT = PI INVERSE), = (3 IF ROT = ROT. TRANS.)	C 3021
C	ITYPE = 1,2,..12 (TYPE OF EULER ANGLE PERMUTATION, SEE IP BELOW)	0 3022
C	ANG = INPUT EULER ANGLES	0 3023
C	ROT = EITHER PI INVERSE OR ROT. TRANS.	0 3024
C		0 3025
C	SEE VOL 1 APPENDIX B FOR THEORETICAL DEVELOPMENT	0 3026
C		0 3027
C	CODED BY CARL BODLEY, NCV. 20, 1973.	0 3028
C	ADDITIONS MADE BY CARL BODLEY, APR. 5, 1974	0 3029
C		0 3030
	DATA IP / 1,2,3, 1,2,1, 1,3,1, 1,3,2,	0 3031
	* 2,3,1, 2,3,2, 2,1,2, 2,1,3,	0 3032
	* 3,1,2, 3,1,3, 3,2,3, 3,2,1 /,	0 3033
	* NOT. EPS / 6, 1.0-06/	0 3034
C		0 3035
	LI = 3*(ITYPE - 1)	0 3036
C		0 3037
	DO 15 L=1,3	0 3038
	IV(L) = 1	0 3039
	DO 10 I=1,3	0 3040
	DO 5 J=1,3	0 3041
	5 A(I,J,L) = ZRO	0 3042
	10 A(I,I,L) = ONE	0 3043
	15 CONTINUE	0 3044
C		0 3045
	LR = 4 - I23	0 3046
	DO 20 L=LR,3	0 3047
	K = IP(LI + L)	0 3048
	S = CSIN(ANG(L))	0 3049
	C = CCOS(ANG(L))	0 3050
	DO TC (1,2,3), K	0 3051
	1 A(2,2,L) = C	0 3052
	A(3,3,L) = C	0 3053
	A(2,3,L) = -S	0 3054
	A(3,2,L) = S	0 3055
	DO TC 20	0 3056
	2 A(1,1,L) = C	0 3057
	A(3,3,L) = C	0 3058

A(1,3,L) = S	C 3059
A(3,1,L) = -S	0 3060
GO TO 20	0 3061
3 A(1,1,L) = C	0 3062
A(2,2,L) = C	0 3063
A(1,2,L) = -S	0 3064
A(2,1,L) = S	C 3065
20 CONTINUE	0 3066
IF (123 .EQ. 1) GO TO 50	0 3067
C	0 3068
C DROP HERE IF ROTATION TRANSFORMATION MATRIX NEEDED	0 3069
C OBTAIN VIA MATRIX PRODUCTS OF ELEMENTARY ROTATION MATRICES	0 3070
CALL MULT3 (A(1,1,1),A(1,1,2),A(1,1,4),3,3,3,3,3)	C 3071
CALL MULT3 (A(1,1,4),A(1,1,3),ROT,3,3,3,3,3)	0 3072
RETURN	0 3073
C	0 3074
C COME HERE TO GET PI INVERSE VELOCITY TRANSFORMATIONS	0 3075
50 DO 25 I=2,3	0 3076
I1 = IP(L1 + I)	0 3077
25 IV(I1) = 1	0 3078
S2 = DSIN(ANG(2))	0 3079
C2 = DCOS(ANG(2))	0 3080
I1 = FLGAT(ITYPE)/4.2	0 3081
K = ITYPE - 4*I1	C 3082
GO TO (61,62,63,64), K	0 3083
61 IF (CABS(C2) .LT. EPS) GO TO 999	0 3084
AL = C2	0 3085
AL2 = AL*AL	C 3086
EE = S2	0 3087
ALP = S2/AL2	0 3088
EEP = -CNE/AL2	0 3089
GO TO 65	0 3090
62 IF (CABS(S2) .LT. EPS) GO TO 999	0 3091
AL = S2	0 3092
AL2 = AL*AL	0 3093
EE = C2	0 3094
ALP = -C2/AL2	0 3095
EEP = CNE/AL2	0 3096
GO TO 65	0 3097
63 IF (CABS(S2) .LT. EPS) GO TO 999	0 3098
AL = -S2	0 3099
AL2 = AL*AL	0 3100
EE = C2	0 3101
ALP = C2/AL2	0 3102
EEP = -CNE/AL2	0 3103
GO TO 65	0 3104
64 IF (CABS(C2) .LT. EPS) GO TO 999	0 3105
AL = C2	0 3106
AL2 = AL*AL	0 3107
EE = -S2	C 3108
ALP = S2/AL2	0 3109
EEP = CNE/AL2	0 3110
C	0 3111
65 K = IP(L1+3)	0 3112
GO TO (71,72,73), K	0 3113
71 A(2,2,1) = A(2,3,3)	0 3114
A(3,3,1) = A(2,3,3)	0 3115
A(3,2,1) = A(2,2,3)	0 3116
A(2,3,1) = -A(2,2,3)	0 3117
A(1,1,1) = 2RD	0 3118

GO TC 75	0 3119
72 A(1,1,1) = A(3,1,3)	0 3120
A(3,3,1) = A(3,1,3)	0 3121
A(1,3,1) = A(1,1,3)	0 3122
A(3,1,1) = -A(1,1,3)	0 3123
A(2,2,1) = ZRO	0 3124
GO TC 75	0 3125
73 A(1,1,1) = A(1,2,3)	0 3126
A(2,2,1) = A(1,2,3)	0 3127
A(2,1,1) = A(1,1,3)	0 3128
A(1,2,1) = -A(1,1,3)	0 3129
A(3,3,1) = ZRO	0 3130
75 DO 8C I=1,3	0 3131
II = IV(1)	0 3132
DO 8C J=1,3	0 3133
A(II,J,2) = A(1,J,1)	0 3134
8C A(II,J,4) = A(1,J,3)	0 3135
C	0 3136
DO 9C J=1,3	0 3137
ROT(1,J) = A(1,J,4)/AL	0 3138
RP3(1,J) = A(1,J,2)/AL	0 3139
ROT(2,J) = A(2,J,4)	0 3140
RP3(2,J) = A(2,J,2)	0 3141
ROT(3,J) = A(3,J,4) - BE*ROT(1,J)	0 3142
RP3(3,J) = A(3,J,2) - BE*RP3(1,J)	0 3143
RP2(1,J) = ALP*A(1,J,4)	0 3144
RP2(2,J) = ZRO	0 3145
9C RP2(3,J) = BEP*A(1,J,4)	0 3146
C	0 3147
RETURN	0 3148
999 WRITE (NOT,1000) ITYPE,ANG(2)	0 3149
1000 FORMAT (1H1,22HGIMBAL LOCK-- ITYPE = ,15,8HANGLE = ,J15.8)	0 3150
STOP	0 3151
END	0 3152
SUBROUTINE UNITY (Z,N,KR)	0 3153
C	0 3154
IMPLICIT REAL*8(A-H,O-Z)	0 3155
DIMENSION Z(KR,1)	0 3156
C	0 3157
C GENERATE A UNITY MATRIX. (ONES ON THE DIAGONAL).	0 3158
C CCOED BY RL WOHLER. FEB 1965.	0 3159
C	0 3160
C SUBROUTINE ARGUMENTS	0 3161
C Z = OUTPUT MATRIX GENERATED. SIZE(N,N).	0 3162
C N = INPUT SIZE OF MATRIX Z (SQUARE).	0 3163
C KR = INPUT ROW DIMENSION OF MATRIX Z IN CALLING PROGRAM.	0 3164
C	0 3165
DO 2C I=1,N	0 3166
DO 1C J=1,N	0 3167
1C Z(I,J) = 0.0 0	0 3168
20 Z(I,I) = 1.0 0	0 3169
RETURN	0 3170
END	0 3171

SUBROUTINE MULTA (AZ,B,NRA,NRB,NCB,KAZ,KB)	0 3172
C	0 3173
IMPLICIT REAL*8(A-H,D-Z)	0 3174
DIMENSION AZ(KAZ,1), B(KB,1),W(150)	70 3175
DATA NOT / 6/	0 3176
C	0 3177
C MATRIX MULTIPLICATION. A * B = Z.	0 3178
C USES TWO WORK SPACES. RESULT (Z) IS PLACED IN A.	0 3179
C AZ MUST BE DIMENSIONED LARGE ENOUGH IN MAIN PROGRAM TO CONTAIN THE	0 3180
C LARGER OF A OR Z.	0 3181
C DEVELOPED BY C S BODLEY. JANUARY 1965.	0 3182
C LAST REVISION BY R F HRUDA. JUNE 1972.	0 3183
C	0 3184
C SUBROUTINE ARGUMENTS	0 3185
C AZ = INPUT MATRIX. SIZE(NRA,NRB).	0 3186
C = OUTPUT RESULT MATRIX. SIZE(NRA,NCB).	0 3187
C B = INPUT MATRIX. SIZE(NRB,NCB)	0 3188
C NRA = INPUT NUMBER OF ROWS OF MATRICES A,Z.	0 3189
C NRNB = INPUT NUMBER OF ROWS OF MATRIX B, COLS OF MATRIX A. MAX=500.	0 3190
C NCB = INPUT NUMBER OF COLS OF MATRICES B,Z.	0 3191
C KAZ = INPUT ROW DIMENSION OF AZ IN CALLING PROGRAM.	0 3192
C KB = INPUT ROW DIMENSION OF B IN CALLING PROGRAM.	0 3193
C	0 3194
IF (NRB .GT. 150) GO TO 999	71 3195
C	0 3196
DO 40 I=1,NRA	0 3197
DO 20 K=1,NRB	0 3198
20 W(K) = AZ(I,K)	0 3199
DO 40 J=1,NCB	0 3200
S = 0.0	0 3201
DO 30 K=1,NRB	0 3202
30 S = S + W(K)*B(K,J)	0 3203
40 AZ(I,J) = S	0 3204
RETURN	0 3205
C	0 3206
999 WRITE (NCT,1001)	0 3207
1001 FORMAT (1H1,31HERROR IN MULTA. PROGRAM STOPPED)	0 3208
STOP	0 3209
END	0 3210

SUBROUTINE BTABA (AZ,B,NRB,NCB,KAZ,KB)	0 3211
C	0 3212
IMPLICIT REAL*8(A-H,D-Z)	0 3213
DIMENSION AZ(KAZ,1), B(KB,1),W(150)	72 3214
DATA NOT / 6/	0 3215
C	0 3216
C TRIPLE MATRIX PRODUCT. B(TRANPOSE) * A * B = Z.	0 3217
C A MUST BE SYMMETRIC TO GET CORRECT ANSWER.	0 3218
C Z WILL BE SYMMETRIC. UPPER HALF CALCULATED, REFLECTED TO LOWER HALF.	0 3219
C USES TWO WORK SPACES. RESULT (Z) IS PLACED IN A.	0 3220
C AZ MUST BE DIMENSIONED LARGE ENOUGH IN MAIN PROGRAM TO CONTAIN THE	0 3221
C LARGER OF A OR Z.	0 3222
C DEVELOPED BY W A BENFIELD. MAY 1972.	0 3223
C LAST REVISION BY R A PHILIPPUS. JUNE 1972.	0 3224
C MODIFIED FOR USE IN GSFC PROGRAM BY CARL BODLEY. MAY 1974	0 3225

C		0 3226
C	SUBROUTINE ARGUMENTS	0 3227
C	AZ = INPUT INNER MATRIX. SIZE(NRB,NRB).	0 3228
C	= OUTPUT RESULT MATRIX. SIZE(NCB,NCB).	0 3229
C	B = INPUT OUTER MATRIX. SIZE(NRB,NCB).	0 3230
C	NRB = INPUT NUMBER OF ROWS OF MATRIX B. SIZE OF MATRIX A. MAX=150.	0 3231
C	NCB = INPUT NUMBER OF COLS OF MATRIX B. SIZE OF MATRIX Z. MAX=150.	0 3232
C	KAZ = INPUT ROW DIMENSION OF AZ IN CALLING PROGRAM.	0 3233
C	KE = INPUT ROW DIMENSION OF B IN CALLING PROGRAM.	0 3234
C		0 3235
	IF (NRB.GT.150 .OR. NCB.GT.150 .OR. NRB.GT.KAZ .OR. NCB.GT.KAZ)	73 3236
	* GC TO 999	0 3237
C		0 3238
	DO 20 I=1,NRB	0 3239
	DO 5 K=1,NRB	0 3240
	5 W(K) = AZ(I,K)	0 3241
	DO 20 J=1,NCB	0 3242
	S = 0.0 0	0 3243
	DO 10 K=1,NRB	0 3244
	10 S = S + W(K)*B(K,J)	0 3245
	20 AZ(I,J) = S	0 3246
C		0 3247
	DO 30 J=1,NCB	0 3248
	DO 25 I=1,J	0 3249
	W(I) = 0.0 0	0 3250
	DO 25 K=1,NRB	0 3251
	25 W(I) = W(I)+B(K,I)*AZ(K,J)	0 3252
	DO 30 I=1,J	0 3253
	AZ(I,J) = W(I)	0 3254
	30 AZ(J,I) = W(I)	0 3255
	RETURN	0 3256
C		0 3257
	999 WRITE (NCT,1001)	0 3258
	1001 FORMAT (1H1,31HERROR IN BTABA. PROGRAM STOPPED)	0 3259
	STOP	0 3260
	END	0 3261
	SUBROUTINE PR3 (A,B,C,W,Z,S,NRA,NCA,NCB,NCC,KA,KB,KC,KW,KZ)	0 3262
C		0 3263
C		0 3264
	IMPLICIT REAL*8(A-H,O-Z)	0 3265
	DIMENSION A(KA,1),B(KB,1),C(KC,1),W(KW,1),Z(KZ,1)	0 3266
CC		0 3267
C	W = A*B	0 3268
CC		0 3269
	DO 10 I=1,NRA	0 3270
	DO 10 J=1,NCB	0 3271
	W(I,J) = 0.0 0	0 3272
	DO 10 K=1,NCA	0 3273
	10 W(I,J) = W(I,J) + A(I,K)*B(K,J)	0 3274
CC		0 3275
C	T	0 3276
C	Z = Z + S*C*W	0 3277
CC		0 3278
	DO 20 I=1,NCC	0 3279

CG 2C J=1,NCB	0 3280
CG 2C K=1,NKA	0 3281
2C Z(I,J) = Z(I,J) + S*C(K,I)*W(K,J)	0 3282
C	0 3283
RETURN	0 3284
END	0 3285
SUBROUTINE DYN320	0 3286
C	0 3287
IMPLICIT REAL*8(A-H,O-Z)	0 3288
C	0 3289
COMMON /AMODW /	0 3290
* AMU(22,22, 6),BW(30,113)	1 3291
COMMON /CCNPAR/	0 3292
* CNTOTA(150)	95 3293
COMMON /DNAUX /	0 3294
* NAUX	0 3295
COMMON /HANDS /	0 3296
* HATH(3,12,10),SIGH(3,12,10),HATS(3,12,15),SIGS(3,12,15)	4 3297
COMMON /ILINER/	0 3298
* IFLNER	0 3299
COMMON /INTGRL/	0 3300
* AM(171, 6),ACCF(9,12, 6),BCCF(6,12, 6),	5 3301
* COF11(12,12, 6),COF22(12,12, 6),COF33(12,12, 6),AK(12,12, 6),	6 3302
* COF12(12,12, 6),COF13(12,12, 6),COF23(12,12, 6),AD(12,12, 6),	7 3303
* COFXY(12,12, 6),COFXZ(12,12, 6),COFYZ(12,12, 6)	8 3304
COMMON /MAXMOM/	0 3305
* NBMAX,NHMAX,NSPMAX,NMWMAX,NMWBOD,NMDBOD,KMU,KY,KU	0 3306
COMMON /MISCNO/	0 3307
* NOPRNT,NOPLOT	0 3308
COMMON /NUMBRS/	0 3309
* ZRU,ONE,TWO,THRE	0 3310
COMMON /PLTOTA/	0 3311
* NRPLUT,NCPLUT	0 3312
COMMON /QPRKTA/	0 3313
* QRK(250),PRK(4), NT	15 3314
COMMON /SPLCIF/	0 3315
* BETAH(6, 6),BLTAHD(6, 6),AMO(2, 5),RH(3,3,30),RS(3,3,30),	16 3316
* CH(3,35),DS(3,30),IMU(3, 5),NMOW(6, 6),IFTSMW(15),	17 3317
* NB,NH,NSPT,NDFMO,NDELTA,ITCPOL(2, 6),IRGF_X(6),IHDATA(7, 6),	18 3318
* LCC(14),LENU(14),NU,NBETA,NLAM,NEQ	19 3319
COMMON /TAPENU/	0 3320
* NTAPE1,NTAPE2,NTAPE3	0 3321
COMMON /TIMESS/	0 3322
* STARTT,DELTAT,T,ENDT,TMST	0 3323
COMMON /VECTOR/	0 3324
* Y(250),YDT(250)	20 3325
COMMON /VINDEP/	0 3326
* INDEP(250)	21 3327
C	0 3328
DATA NJT / 6/	0 3329
C	0 3330
2001 FORMAT (////IX,47HTHE FOLLOWING INTEGER ARRAY (INDEP) PRESCRIBES	0 3331
*54HINDEPENDENT VARIABLES (1), AND DEPENDENT VARIABLES (0), /IX,	0 3332
* 101(1H-))	0 3333

C		0 3334
	PRK(1) = ONE/TWO	0 3335
	PRK(2) = ONE - DSORT{PRK(1)}	0 3336
	PRK(3) = TWO - PRK(2)	0 3337
	PRK(4) = PRK(1)	0 3338
C		0 3339
	NT = 0	0 3340
	T = STARTT	0 3341
	TMST = ZRO	0 3342
C		0 3343
	DO 100 I=1,NEQ	0 3344
100	CRK(I) = ZRO	0 3345
C		0 3346
C		0 3347
	REWIND NTAP1	0 3348
C		0 3349
	DO 2 I=1,KY	0 3350
	INDEF(I) = 1	0 3351
	Y(I) = ZRO	0 3352
2	YDT(I) = ZRO	0 3353
C		0 3354
	DO 3 N=1,NB	0 3355
	NXE = IRGFLX(N)	0 3356
	NP6 = 6 + NXE	0 3357
C	READ BODY N INERTIA MATRIX	0 3358
	READ (NTAPE1) ((AMU(I,J,N),J=1,NP6),I=1,NP6)	0 3359
C	LOAD UNDEFORMED INERTIA MATRIX INTO THE AM ARRAY.	0 3360
C	HEREAFTER AMU ARRAY WILL CONTAIN THE DEFORMED INERTIA MATRICES	0 3361
	KNT = J	0 3362
	DO 6 I=1,NP6	0 3363
	DO 6 J=1,NP6	0 3364
	KNT = KNT + 1	0 3365
6	AM(KNT,N) = AMU(I,J,N)	0 3366
C		0 3367
C	DO TC 3 IF BODY N A RIGID BODY	0 3368
	IF (NXE .EQ. 0) GO TO 3	0 3369
C		0 3370
C		0 3371
	LXE = LCCU(N+NB)	0 3372
	LXED = LCCU(N) + 6	0 3373
	LXEN = LXE + NXE - 1	0 3374
	LXEDN = LXED + NXE - 1	0 3375
C		0 3376
C	ACCF = ALPHA COEFFICIENTS OF EQ II-88 USED TO DEFINE BODY'S	0 3377
C	STATIC MASS MOMENTS LINEAR DEPENDENCE ON DEFORMATION	0 3378
C	BCCF = B COEFFICIENTS OF EQ II-88 USED TO DEFINE BODY INERTIA	0 3379
C	TENSOR'S LINEAR DEPENDENCE ON DEFORMATION	0 3380
C	CCFXY = C SUB XY COEFFICIENTS OF EQ II-89	0 3381
C	COFXZ = C SUB XZ COEFFICIENTS OF EQ II-89	0 3382
C	CCFYZ = C SUB ZY COEFFICIENTS OF EQ II-89	0 3383
C	CGF11 = C SUB 11 COEFFICIENTS OF EQ II-89, INERTIA TENSOR QUAD DEP	0 3384
C	COF22 = C SUB 22 COEFFICIENTS OF EQ II-89, INERTIA TENSOR QUAD DEP	0 3385
C	COF33 = C SUB 33 COEFFICIENTS OF EQ II-89, INERTIA TENSOR QUAD DEP	0 3386
C	CCF12 = C SUB 12 COEFFICIENTS OF EQ II-89, INERTIA TENSOR QUAD DEP	0 3387
C	CGF13 = C SUB 13 COEFFICIENTS OF EQ II-89, INERTIA TENSOR QUAD DEP	0 3388
C	COF23 = C SUB 23 COEFFICIENTS OF EQ II-89, INERTIA TENSOR QUAD DEP	0 3389
	READ (NTAPE1) ((ACOF(I,J,N),J=1,NXE),I=1,9)	0 3390
	READ (NTAPE1) ((BCOF(I,J,N),J=1,NXE),I=1,6)	0 3391
	READ (NTAPE1) ((COFXY(I,J,N),J=1,NXE),I=1,NXE)	0 3392
	READ (NTAPE1) ((COFXZ(I,J,N),J=1,NXE),I=1,NXE)	0 3393

READ (NTAPE1) ((COFYZ(1,J,N),J=1,NXE),I=1,NXE)	0 3394
READ (NTAPE1) ((CUF11(1,J,N),J=1,NXE),I=1,NXE)	0 3395
READ (NTAPE1) ((CUF22(1,J,N),J=1,NXE),I=1,NXE)	0 3396
READ (NTAPE1) ((CUF33(1,J,N),J=1,NXE),I=1,NXE)	0 3397
READ (NTAPE1) ((CUF12(1,J,N),J=1,NXE),I=1,NXE)	0 3398
READ (NTAPE1) ((CUF13(1,J,N),J=1,NXE),I=1,NXE)	0 3399
READ (NTAPE1) ((CUF23(1,J,N),J=1,NXE),I=1,NXE)	0 3400
C	0 3401
C AK(N) MODAL STIFFNESS MATRIX BODY N	0 3402
C AC(N) MODAL DAMPING MATRIX BODY N	0 3403
READ (NTAPE1) ((AK (1,J,N),J=1,NXE),I=1,NXE)	0 3404
READ (NTAPE1) ((AD (1,J,N),J=1,NXE),I=1,NXE)	0 3405
C	0 3406
C INITIAL CONDITIONS FOR FLEXIBLE BODY COORDINATES READ FROM	0 3407
C TAPE DIRECTLY INTO STATE VECTOR	0 3408
READ (NTAPE1) (Y(J),J=LXE ,LXEN)	0 3409
READ (NTAPE1) (Y(J),J=LXED,LXEDN)	0 3410
C	0 3411
READ (NTAPE1) NHB	0 3412
CCC NOTE --- NHB IS NO. OF P/Q HINGES, NOT TO INCLUDE HINGE-1.	0 3413
CCC OVERLAY(1,0) MUST BE SURE OF THIS.	0 3414
DO 4 L=1,NHB	0 3415
READ (NTAPE1) NOH	0 3416
CCCC NCH NOT TO INCLUDE HINGE 1 ****	0 3417
LHS = 0	0 3418
IF (ITOPCL(1,NOH) .EQ. N) LHS = 2*NOH - 3	0 3419
IF (ITOPCL(2,NOH) .EQ. N) LHS = 2*NOH - 2	0 3420
IF (LHS .LE. 0) GO TO 999	0 3421
READ (NTAPE1) ((HATH(1,J,LHS),J=1,NXE),I=1,3)	0 3422
4 READ (NTAPE1) ((SIGH(1,J,LHS),J=1,NXE),I=1,3)	0 3423
READ (NTAPE1) NSB	0 3424
IF (NSB .EQ. 0) GO TO 3	0 3425
DO 5 L=1,NSB	0 3426
READ (NTAPE1) NUS	0 3427
READ (NTAPE1) ((HATS(1,J,NUS),J=1,NXE),I=1,3)	0 3428
5 READ (NTAPE1) ((SIGS(1,J,NUS),J=1,NXE),I=1,3)	0 3429
3 CONTINUE	0 3430
C	0 3431
C	0 3432
C LOAD INITIAL MOMENTUM WHEEL RATES IN STATE VECTOR	0 3433
DO 10 N=1,NB	0 3434
LTC = LDCU(N) + IRCFLX(N) + 5	0 3435
NMW = NMCW(1,N)	0 3436
IF (NMW .EQ. 0) GO TO 10	0 3437
NMWVS = 0	0 3438
DO 15 I=1,NMW	0 3439
NOMW = NNOW(I+2,N)	0 3440
IF (IMC(3,NOMW) .EQ. 0) GO TO 15	0 3441
NMWVS = NMWVS + 1	0 3442
Y(LTC+NMWVS) = AMO(1,NOMW)	0 3443
15 CONTINUE	0 3444
10 CONTINUE	0 3445
C	0 3446
C LOAD INITIAL HINGE ANGLES AND DISPLACEMENTS IN STATE VECTOR	0 3447
LBE = LDCU(2*NB+1) - 1	0 3448
DO 20 J=1,NH	0 3449
DO 20 I=1,0	0 3450
IPI = I + 1	0 3451
IF (IRDATA(IPI,J) .EQ. 1) GO TO 20	0 3452
LBE = LBE + 1	0 3453

Y(LBE) = BETAH(I,J)	0 3454
20 CONTINUE	0 3455
C	0 3456
C LOAD INITIAL VALUES FOR CONTROL SYSTEM VARIABLES IN STATE VECTOR	0 3457
IF (NDELTA .EQ. 0) GO TO 25	0 3458
LDE = LCCU(2*NB+2) - 1	0 3459
GO 26 J=1,NDELTA	0 3460
L = LDE + J	0 3461
26 Y(L) = CNTDTA(J)	0 3462
C	0 3463
C BRANCH TO ROTDH TO GET RELATIVE ORIENTATION AND POSITION	0 3464
C DATA FOR CONTIGUOUS BODIES AND BODY FIXED FRAMES RELATIVE TO	0 3465
C INERTIAL REFERENCE	0 3466
25 CALL ROTDH	0 3467
C BRANCH TO BHGENR TO COMPUTE HINGE POINT KINEMATICS	0 3468
CALL BHGENR	0 3469
C BRANCH TO FINDU TO GET INDEPENDENT COMPONENTS OF U VECTOR	0 3470
C FOR INSERTION INTO THE STATE VECTOR Y	0 3471
CALL FINDU (0)	0 3472
C	0 3473
IPRNT = 0	0 3474
IPLT = 0	0 3475
C	0 3476
C BRANCH TO YDOT TO GET TIME DERIVATIVE OF STATE VECTOR, RESULTS	0 3477
C STORED IN YDT ARRAY EQUATIONS II-1 THRU 6 ALL EVALUATED IN YDOT	0 3478
CALL YDOT	0 3479
WRITE (NGT,2001)	0 3480
CALL WRITIS (INDEP,1,NEQ,1)	0 3481
C	0 3482
NCAM = NEQ	0 3483
NRAM = NCAM + NAUX	0 3484
C	0 3485
C COMPUTE ENERGY AND MOMENTUM FOR BODIES AND SYSTEM	0 3486
CALL ENGMOM	0 3487
CALL PRNTOU	0 3488
IF (IFLNER .EQ. 1) GO TO 50	0 3489
CALL PLCTWR	0 3490
C	0 3491
C START INTEGRATION OF EQUATIONS OF MOTION	0 3492
C NEQ = TOTAL NUMBER OF EQUATIONS TO INTEGRATE	0 3493
C RETURN TO THIS POINT AFTER EACH INTEGRATION CYCLE	0 3494
11 CALL RKADAM(NEQ)	0 3495
CALL ENGMOM	0 3496
IPRNT = IPRNT + 1	0 3497
IF (IPRNT .NE. NOPRNT) GO TO 12	0 3498
CALL PRNTOU	0 3499
IPRNT = 0	0 3500
12 CONTINUE	0 3501
IPLT = IPLT + 1	0 3502
IF (IPLT .NE. NOPLOT) GO TO 13	0 3503
CALL PLCTWR	0 3504
IPLT = 0	0 3505
13 CONTINUE	0 3506
IF (1 .LT. ENDT) GO TO 11	0 3507
RETURN	0 3508
C	0 3509
C COME HERE IF EQUATIONS TO BE LINEARIZED	0 3510
50 CALL LINEAR (NRAM,NCAM)	0 3511
RETURN	0 3512
C	0 3513

599	WRITE (NUT,2010)	C 3514
2010	FORMAT (I11,15HERROR IN ITCPOL)	C 3515
	STOP	0 3516
	END	0 3517
	SUBROUTINE FINDU (IFLAG)	0 3518
C	IMPLICIT REAL*8(A-H,O-Z)	0 3519
	DEBUG # 38	0 3520
C		0 3521
	DIMENSION ICON(36), IVEC(36), ICCN(113), R(36), JBIGST(6), JBIGFN(6)	74 3522
C		0 3523
	COMMON /AMUB*/	0 3524
*	AMU(22,22, 6),BW(36,113)	1 3525
	COMMON /BHESRD/	0 3526
*	BH(6,18,11),BS(6,18,15),ROL(3,3, 6),JUL(3, 6)	2 3527
	COMMON /NUMBRS/	0 3528
*	ZRC,ONE,TWO,THREE	0 3529
	COMMON /SPECIF/	0 3530
*	BETAH(6, 6),BETAHD(6, 6),AMO(2, 5),RH(3,3,30),RS(3,3,30),	16 3531
*	BH(3,35),DS(3,30),IMU(3, 5),NMCW(6, 5),IFTSMA(15),	17 3532
*	NU,NH,NSPT,NDFMO,NDELTA,ITCPOL(2, 6),IRGFLX(6),IHDATA(7, 6),	18 3533
*	LUCL(14),LENU(14),NU,NBETA,NLAM,NEQ	19 3534
	COMMON /TIMESS/	0 3535
*	STARTT,DELTA,T,ENDT,IMST	0 3536
	COMMON /VECTOR/	0 3537
*	Y(250),YOT(250)	20 3538
	COMMON /VINDEP/	0 3539
*	INDEP(250)	21 3540
	COMMON /MAXIMUM/	0 3541
*	NJMAX,NHMAX,NSPMAX,NM*MAX,NM*BDU,NM*BJJ,K*U,K*Y,K*U	0 3542
C		0 3543
	DATA EPS,NOT / 1.D-06, 6/	0 3544
C		0 3545
	LOGICAL DEBUG,LEQU	0 3546
	COMMON /LDEBUG/ DEBUG(120)	0 3547
	EQUIVALENCE (LEQU,DEBUG(38))	0 3548
	IF(LEQU) WRITE(NUT,2011) IFLAG	0 3549
2011	FORMAT (' SUBROUTINE FINDU(' ,I1,')')	0 3550
	IF (IFLAG.EQ. 2) GO TO 100	0 3551
	IF (IFLAG.EQ. 1) GO TO 110	0 3552
C		0 3553
C	INITIAL PASS THROUGH IFLAG=0	0 3554
C	IFLAG = 0 INITIAL PASS THROUGH GET INDEPENDENT COORDINATES	0 3555
C	TO PUT IN STATE VECTOR Y	0 3556
C	IFLAG = 1 NOT AN INTERMEDIATE STEP THROUGH RUNGE KUTTA	0 3557
C	GET DEPENDENT COORDINATES FOR INCLUSION IN Y ARRAY	0 3558
C	IFLAG = 2 AN INTERMEDIATE STEP THROUGH RUNGE KUTTA	0 3559
C	USE SAME SET OF INDEPENDENT COORDINATES AS USED IN IFLAG=1	0 3560
C		0 3561
C	IFLAG=0 SETUP	0 3562
	NCN = 6*NH	0 3563
	DO 205 L=1,NCN	0 3564
	IVEC(L) = 0	0 3565
205	ICCN(L) = 1	0 3566
	DO 207 J=1,NU	0 3567

207	JCCN(J) = 0	0 3568
	DO 210 N=1,NB	0 3569
	JU = LOCL(N) - 1	0 3570
C	JRNG > 6 ONLY FOR NON-TOPOLOGICAL TREES. FLEXIBLE BODY	0 3571
C	COORDINATES NEEDED TO AVOID STATIC INDETERMINANCY	0 3572
	JRNG = 6	0 3573
	IF (NH .GT. NB) JRNG = 6 + IRGFLX(N)	0 3574
	DO 210 J=1,JRNG	0 3575
210	JCCN(JU+J) = 1	0 3576
	NR = 0	0 3577
	DO 215 J=1,NH	0 3578
	DO 215 I=1,6	0 3579
	IPI = I + 1	0 3580
	NR = NR + 1	0 3581
C	LOAD R ARRAY WITH INITIAL BETAHD ARRAY	0 3582
	R(NR) = BETAHD(I,J)	0 3583
C	CHECK FOR FIXED OR RHEUMIC CONSTRAINT CONDITIONS	0 3584
	IF (IHDATA(IPI,J) .EQ. 1) R(NR) = ZRO	0 3585
	IF (IHDATA(IPI,J) .EQ. 2) R(NR) = ADT(NR,T)	0 3586
	BETAHD(I,J) = R(NR)	0 3587
215	CONTINUE	0 3588
C		0 3589
	GO TO 150	0 3590
C		0 3591
C	COME HERE TO GET CONSTRAINT EQUATIONS SETUP	0 3592
C	IFLAG = 1	0 3593
C	START SEARCH FOR NEW AND POSSIBLY BETTER SET OF	0 3594
C	INDEPENDENT COORDINATES(ONLY ON LAST PASS THRU RUNGE KUTTA)	0 3595
110	DO 220 L=1,NCN	0 3596
	IVC(L) = 0	0 3597
220	ICCN(L) = 0	0 3598
	DO 221 J=1,NU	0 3599
221	INDEF(J) = 1	0 3600
C	CONSTRAINT INFO INTO ICON	0 3601
	NR = 0	0 3602
	DO 225 J=1,NH	0 3603
	DO 225 I=2,7	0 3604
	NR = NR + 1	0 3605
	ICCN(NR) = IHDATA(I,J)	0 3606
225	CONTINUE	0 3607
C		0 3608
C	IFLAG = 2	0 3609
C	USE SAME SET OF INDEPENDENT COORDINATES AS IN IFLAG=1 PASS	0 3610
100	DO 230 L=1,NCN	0 3611
C	LOAD CONSTRAINT DATA INTO R ARRAY	0 3612
	IF (ICCN(L) .EQ. 0) GO TO 230	0 3613
	IF (ICCN(L) .EQ. 1) R(L) = ZRO	0 3614
	IF (ICCN(L) .EQ. 2) R(L) = ADT(L,T)	0 3615
230	CONTINUE	0 3616
C		0 3617
C	LOAD BW ARRAY WITH ELEMENTS OF BH ARRAY COMPUTED IN BHGENER	0 3618
C	BW = COEFFICIENT MATRIX OF EQUATION 11-84	0 3619
150	DO 310 I=1,NCN	0 3620
	DO 310 J=1,NU	0 3621
310	BW(I,J) = ZRO	0 3622
	LEC = 6 + IRGFLX(1)	0 3623
	JBIGST(1) = 1	0 3624
	JBIGFN(1) = LEC	0 3625
	DO 315 I=1,6	0 3626
	DO 315 J=1,LEC	0 3627

315	EW(I,J) = BH(I,J,1)	0 3628
C		0 3629
	CO 320 L=2,NH	0 3630
	LQ = 2*L - 2	0 3631
	LP = LQ + 1	0 3632
	NOBQ = ITOPUL(1,L)	0 3633
	NOBP = ITOPUL(2,L)	0 3634
	LH = 6*(L-1)	0 3635
	LBQ = LCCU(NOBBQ) - 1	0 3636
	LBP = LCCU(NOBBP) - 1	0 3637
	LEG = 6 + IRGFLX(NOBBQ)	0 3638
	LEP = 6 + IRGFLX(NOBBP)	0 3639
	JBIGST(L) = LBQ+1	0 3640
	JBIGFN(L) = LBQ+LEG	0 3641
	CO 325 I=1,6	0 3642
	CO 325 J=1,LEQ	0 3643
325	EW(I+LH,J+LBQ) = BH(I,J,LQ)	0 3644
	CO 330 I=1,6	0 3645
	CO 330 J=1,LEP	0 3646
330	EW(I+LH,J+LBP) = BH(I,J,LP)	0 3647
320	CONTINUE	0 3648
C		0 3649
	IF(.NOT.LEQU) GO TO 70	0 3650
	WRITE(NC,1002)	0 3651
1002	FORMAT (' THE BW MATRIX IS')	0 3652
	NH6 = NHMAX*6	0 3653
	CALL WRITES(BW,NCN,NU,NH6)	0 3654
70	CONTINUE	0 3655
C	IF IFLAG = 0	0 3656
C	USE REDUCTION PROCESS TO SOLVE FOR U AND TO PICK OUT THE	0 3657
C	INDEPENDENT COORDINATES TO BE INSERTED INTO THE STATE VECTOR Y	0 3658
C	IFLAG = 1 OR 2 DEPENDENT COORDINATES IN Y ARRAY COMPUTED	0 3659
C	IFLAG = 1 CHECK FOR BEST SET OF INDEPENDENT COORDINATES	0 3660
C	IFLAG = 2 WITHIN AN INTEGRATION LOOP DON'T CHANGE SET OF	0 3661
C	INDEPENDENT COORDINATES BEING USED	0 3662
C	ANALOGOUS TO SWITCHING EULER ANGLE SEQUENCE WHEN GIMBAL	0 3663
C	LOCK IS APPROACHED	0 3664
C		0 3665
C	PROCEED TO INVERT SUB-PARTITION OF BW ARRAY AND SOLVE FOR	0 3666
C	DEPENDENT COORDINATES IF IFLAG=1 OR 2, INDEPENDENT	0 3667
C	COORDINATES IF IFLAG=0	0 3668
	CO 10 L=1,NCN	0 3669
	IF(10CN(L).EQ.0) GO TO 10	0 3670
	IST = 0	0 3671
	LL = 1 + (L-1)/6	0 3672
	LS = 6*(LL-1) + 3	0 3673
	JST = JBIGST(LL)	0 3674
	IF(L.GT.LS) JST = JST+3	0 3675
	JFN = JBIGFN(LL)	0 3676
400	JBIG = 0	0 3677
	A = ZRO	0 3678
	DO 15 J=JST,JFN	0 3679
	IF (JCUN(J).EQ. 0) GO TO 15	0 3680
	AT = CABS(BW(L,J))	0 3681
	IF (AT .LT. A) GO TO 15	0 3682
	A = AT	0 3683
	JBIG = J	0 3684
15	CONTINUE	0 3685
	IVEC(L) = JBIG	0 3686
	IF(A .GT. EPS) GO TO 400	0 3687

IF (A .LE. EPS.AND. IST.EQ.1) GO TO 999	0 3688
IST = 1	0 3689
JST = 1	0 3690
JFN = NU	0 3691
GO TO 400	0 3692
405 F = EW(L,JBIG)	0 3693
DO 17 J=1,NU	0 3694
17 BW(L,J) = BW(L,J)/F	0 3695
R(L) = R(L)/F	0 3696
EW(L,JBIG) = ONE	0 3697
DO 25 I=1,NCN	0 3698
IF (I .EQ. L .OR. ICUN(I) .EQ. 0) GO TO 25	0 3699
F = EW(I,JBIG)	0 3700
DO 30 J=1,NU	0 3701
30 BW(I,J) = BW(I,J) - F*BW(L,J)	0 3702
R(I) = R(I) - F*R(L)	0 3703
EW(I,JBIG) = ZRO	0 3704
25 CONTINUE	0 3705
10 CONTINUE	0 3706
C	0 3707
C LCAD COORDINATES COMPUTED IN Y ARRAY	0 3708
DO 35 L=1,NCN	0 3709
LU = IVEC(L)	0 3710
IF (LU .EQ. 0) GO TO 35	0 3711
Y(LU) = ZRO	0 3712
35 CONTINUE	0 3713
C	0 3714
DO 40 L=1,NCN	0 3715
IF (ICON(L) .EQ. 0) GO TO 40	0 3716
DO 45 J=1,NU	0 3717
45 R(L) = R(L) - BW(L,J)*Y(J)	0 3718
40 CONTINUE	0 3719
DO 50 L=1,NCN	0 3720
LU = IVEC(L)	0 3721
IF (LU .EQ. 0) GO TO 50	0 3722
C SET	0 3723
C INDEP(LU) = 0	0 3724
C IF COORDINATE LU IN STATE VECTOR IS DEPENDENT	0 3725
IF (IFLAG .EQ. 1) INDEP(LU) = 0	0 3726
Y(LU) = R(L)	0 3727
50 CONTINUE	0 3728
C	0 3729
IF (.NOT. LEQU) RETURN	0 3730
WRITE (NCT,1004)	0 3731
1004 FORMAT (' INDEP ARRAY IS ')	0 3732
CALL WRITIS(INDEP,1,NU,1)	0 3733
RETURN	0 3734
999 WRITE (NCT,1001)	0 3735
1001 FORMAT ('H1,25HSINGULAR EQUATIONS. FINDU')	0 3736
STOP	0 3737
C	0 3738
END	0 3739

SUBROUTINE YDOT	0 3740
C	0 3741
	DEBUG # 39

```

      IMPLICIT REAL*8(A-H,O-Z)
C
      COMMON /AMOB# /
      *      AMU(22,22, 6),BW(36,113)
      COMMON /BHBSRJ/
      *      BH(6,18,11),BS(6,18,15),ROL(3,3, 6),JJE(3, 6)
      COMMON /HANDS /
      *      HATH(3,12,10),SIGH(3,12,10),HATS(3,12,15),SIGS(3,12,15)
      COMMON /ILINER/
      *      IFLNER
      COMMON /INTGRL/
      *      AM(171, 6),ACOF(9,12, 6),BCOF(6,12, 6),
      *      COF11(12,12, 6),COF22(12,12, 6),COF33(12,12, 6),AK(12,12, 6),
      *      CCF12(12,12, 6),COF13(12,12, 6),COF23(12,12, 6),AD(12,12, 6),
      *      COFX1(12,12, 6),COFX2(12,12, 6),COFY2(12,12, 6)
      COMMON /IVCONS/
      *      IV(6, 6)
      COMMON /JILFLG/
      *      JIL
      COMMON /LAMBDA/
      *      ALAM(36)
      COMMON /MAXMUM/
      *      NBMAX,NHMAX,NSPMAX,NMWMAX,NMWRBD,NMDBDD,CMU,KY,KU
      COMMON /MLMENG/
      *      P(113),PMUM(36),HTUT(3),TOTL(3),ENGKE( 6),ENGPE( 6),
      *      TUTKE, TUTPE, TUTENG, AHTEI,ATOTL
      COMMON /NUMBRS/
      *      ZRC,ONE,TWO,THRS
      COMMON /SPECIF/
      *      BETAH(6, 6),BETAHD(6, 6),AMD(2, 5),RH(3,3,30),RS(3,3,30),
      *      CH(3,35),US(3,30),IMU(3, 5),NMCW(6, 6),IFSMW(15),
      *      NJ,NH,NSPT,NCFMG,NDELTA,ITCPQL(2, 6),IRGFLX( 6),IHDATA(7, 6),
      *      LUCL(14),LENU(14),NU,NBETA,NLAM,NEQ
      COMMON /TAPEND/
      *      NTAPE1,NTAPE2,NTAPE3
      COMMON /TIMESS/
      *      STARTT,DELTAT,T,ENDT,TMST
      COMMON /VECTOR/
      *      Y(250),YDT(250)
C
      DIMENSION GGV(113),U(36),V(36),BDTQ(6,18),BDTP(6,18)
C
      DIMENSION BMB(36,36),BM(6,22,11)
      EQUIVALENCE (BW(1),BMB(1)), (BW(1297),BM(1))
      EQUIVALENCE (ALAM(1),V(1))
C
      DATA IFLAG / 1 /
      DATA NOT/0 /
C
      LOGICAL DEBUG,LEQU
      COMMON /LDEBUG/ LDEBUG(120)
      EQUIVALENCE (LEQU,LDEBUG(39))
      IF(LEQU) WRITE(NOT,1000)
1000 FORMAT (' SUBROUTINE YOUT')
      IF (JIL .EQ. 4) IFLAG = 1
C
      IF(.NOT.LEQU) GO TO 21
      WRITE (NOT,1002)
1002 FORMAT (' THE STATE VECTOR Y IS ')
      CALL WRITES(Y,1,NEQ,1)

```

21 CONTINUE	0 3802
C	0 3803
C GET DATA TO DEFINE HINGE AND BODY FIXED FRAME KINEMATICS	0 3804
CALL RUTCH	0 3805
C	0 3806
C GENERATE BH ARRAY	0 3807
CALL BHGENR	0 3808
C	0 3809
C GET DATA TO DEFINE SENSOR POINT AXIS SYSTEM ORIENTATION	0 3810
CALL ROTDS	0 3811
C	0 3812
C GET DATA TO DEFINE KINEMATICAL COEFFICIENTS FOR SENSOR POINTS	0 3813
CALL BSGENR	0 3814
C	0 3815
C BRANCH TO MGEN TO GENERATE DEFORMATION DEPENDENT MASS MATRIX	0 3816
C AND GRAVITY GRADIENT DATA, SEE VOL 1 PAGES 48-52	0 3817
CALL MGEN	0 3818
C	0 3819
C IF NLAM(NUMBER OF CONSTRAINTS) GREATER THAN ZERO BRANCH TO	0 3820
C FINDU(1) TO GET DEPENDENT COORDINATES TO PUT INTO STATE	0 3821
C VECTOR ARRAY Y	0 3822
C IFLAG = 2 ON INTERMEDIATE STEP OF RUNGE KUTTA CYCLE	0 3823
IF (NLAM .GT. 0) CALL FINDU (IFLAG)	0 3824
C	0 3825
DO 10 L=1,NB	0 3826
LO = LDCU(L)	0 3827
LE = LEND(L)	0 3828
C	0 3829
C GET ORDINARY MOMENTA EQUATION II-49 VOL 1	0 3830
10 CALL MULT3 (AMU(1,1,L),Y(LO),P(LO),LE,LE,1,K4J,1,1)	0 3831
DO 11 I=1,NU	0 3832
11 GGV(I) = ZRO	0 3833
C	0 3834
C GET GRAVITY GRADIENT DATA	0 3835
CALL GRVGRD (GGV)	0 3836
DO 15 L=1,NB	0 3837
LE = LEND(L)	0 3838
C	0 3839
C INVERT THE MASS MATRIX FOR BODY L STORE RESULT IN AMU ARRAY	0 3840
C MASS MATRIX DESTROYED HERE	0 3841
15 CALL INVINP (AMU(1,1,L),AMU(1,1,L),LE,KMU)	0 3842
IF (NLAM .EQ. 0) GO TO 200	0 3843
C	0 3844
C GET B*M**(-1)*B**T FOR USE IN OBTAINING LAGRANGE MULTIPLIERS	0 3845
CALL GETEMB	0 3846
KMB = 6*NBMAX	0 3847
C DECOMPOSE BMB MATRIX	0 3848
CALL DCCM2 (BMB,D,NLAM,KMB)	0 3849
C	0 3850
C	0 3851
CC CALCULATE BETADT AND PLACE INTO YDT	0 3852
C COMPUTE BETA OUT TERM OF EQUATION II-3 VOL 1 PUT IN YDT ARRAY	0 3853
C ALSO SAVE IT IN BETAND ARRAY	0 3854
200 IC = LDCU(2*NB+1) - 1	0 3855
DO 60 L=1,NB	0 3856
DO 60 I=1,6	0 3857
IP1 = I + 1	0 3858
IF (IFDATA(IP1,L) .EQ. 1) GO TO 60	0 3859
IC = IC + 1	0 3860
YDT(IC) = ZRO	0 3861

IF (L .EQ. 1) GO TO 61	0 3862
NOBQ = ITCPUL(1,L)	C 3863
NOBP = ITOPUL(2,L)	0 3864
LO = 2*L - 2	0 3865
LP = LO + 1	0 3866
LQC = LCCU(NOBO) - 1	0 3867
LQP = LCCU(NOBP) - 1	0 3868
LEQ = IRGFLX(NOBO) + 6	0 3869
LEP = IRGFLX(NOBP) + 6	0 3870
C	C 3871
C EVALUATE ALL BETA DOT TERMS IN EQUATION 11-3 STACK THEM	0 3872
C IN YDT ARRAY AFTER MODAL VELOCITIES	0 3873
CO 62 J=1,LEQ	0 3874
LOCJ = LQJ + J	0 3875
62 YDT(IC) = YDT(IC) + BH(1,J,LQ)*Y(LOCJ)	0 3876
CO 63 J=1,LEP	0 3877
LOPJ = LCP + J	0 3878
63 YDT(IC) = YDT(IC) + BH(1,J,LP)*Y(LOPJ)	0 3879
EETAFO(1,L) = YDT(IC)	0 3880
GO TO 60	0 3881
61 LEQ = IRGFLX(1) + 6	0 3882
CO 64 J=1,LEQ	0 3883
64 YDT(IC) = YDT(IC) + EH(1,J,1)*Y(J)	0 3884
EETAFO(1,L) = YDT(1,L)	0 3885
60 CONTINUE	0 3886
C	0 3887
C PUT MODAL VELOCITIES (X1(OUT)) INTO YDOT	C 3888
C THESE GO AFTER BODY ACCELERATIONS. THEY CAN BE FOUND IN THE	0 3889
C STATE VECTOR SEE ARRANGEMENT OF U VECTOR	0 3890
C	0 3891
CO 65 N=1,NB	0 3892
LE = IRGFLX(N)	0 3893
IF (LE .EQ. 0) GO TO 65	0 3894
LOU = LCCU(N) + 5	0 3895
LO = LCCU(N+NE) - 1	0 3896
CO 66 J=1,LE	0 3897
66 YDT(LO+J) = Y(LOU+J)	0 3898
65 CONTINUE	0 3899
C	0 3900
C BRANCH TO TORQUE TO COMPUTE ALL TIME DEPENDENT FORCES FOR ALL	0 3901
C BODIES,GGV ARRAY CONTAINS SUM OF ALL FORCES,SEE VOL 1,PP6-8	0 3902
C CALL TORQUE (GGV)	0 3903
C	0 3904
CC INITIALIZE UDOT, GET RHS OF LAMBDA EQUATION	C 3905
C	0 3906
C COMPUTE M**(-1)*G PUT RESULT IN APPROPRIATE ROW OF YDT	C 3907
CO 70 N=1,NB	0 3908
LO = LCCU(N)	0 3909
LE = LENDU(N)	0 3910
70 CALL MULTS (AMU(1,1,N),GGV(LO),YDT(LO),LE,LE,1,KMU,1,1)	C 3911
IF (.NOT.LEQU) GO TO 20	0 3912
WRITE(NDT,1001)	0 3913
1001 FORMAT (' YDT ARRAY SANS CONSTRAINT TORQUES ')	0 3914
CALL WRITES(YDT,1,NEU,1)	C 3915
20 CONTINUE	0 3916
C	0 3917
IFLAG = 2	0 3918
IF (NLAM .EQ. 0) RETURN	0 3919
C	0 3920
C IF FIXED OR RHEONOMIC CONSTRAINTS EXIST EVALUATE ALL LAGRANGE	0 3921

```

C      MULTIPLIERS VIA EQUATION 11-6 VOL 1                                0 3922
DO 80 L=1,NH                                                              0 3923
C      BRANCH TO BDOTP TO GET BDTQ AND BDP. THE LOWER CASE B DOT      0 3924
C      MATRIX IN EQUATION 11-6 VOL 1                                     0 3925
      (CALL BDOTP (L,BDTQ,BDP))                                           0 3926
DO 80 I=1,6                                                                0 3927
      IP1 = I + 1                                                         C 3928
      IC = IV(I,L)                                                         C 3929
      IF (IC.EQ. 0) GO TO 80                                              0 3930
      V(IC) = ZRU                                                         0 3931
      IIC = 0*(L-1) + 1                                                  0 3932
C      IF RHEONOMIC CONSTRAINTS CALL FUNCTION ADDT TO DEFINE TIME      0 3933
C      DEPENDENT ACCELERATION. THE ALPHA DOUBLE DOT TERM IN EQ. 11-6  0 3934
      IF (IFDATA(IP1,L).EQ. 2) V(IC) = ADDT(IIC,I)                      0 3935
      IF (L.EQ. 1) GO TO 81                                              0 3936
      NOBQ = ITOPUL(1,L)                                                0 3937
      NOBP = ITOPUL(2,L)                                                0 3938
      LQ = 2*L - 2                                                       0 3939
      LP = LQ + 1                                                        0 3940
      LOQ = LCU(NOBQ) - 1                                                0 3941
      LQP = LCU(NOBP) - 1                                                0 3942
      LEG = IRGFLX(NOBQ) + 0                                             0 3943
      LEF = IRGFLX(NOBP) + 0                                             0 3944
C      GET REST OF TERMS WITHIN PARENTHESES IN EQUATION 11-6          C 3945
DO 82 J=1,LEQ                                                             0 3946
      LOQJ = LQJ + J                                                    0 3947
82  V(IC) = V(IC) - BH(I,J,LQ)*YDT(LOQJ) - BDTQ(I,J)*Y(LQJ)          0 3948
DO 83 J=1,LEP                                                             0 3949
      LQPJ = LQP + J                                                    0 3950
83  V(IC) = V(IC) - BH(I,J,LP)*YDT(LQPJ) - BDP(I,J)*Y(LPJ)          0 3951
GO TO 80                                                                  0 3952
81  LEG = IRGFLX(1) + 0                                                 0 3953
DO 84 J=1,LEQ                                                             0 3954
84  V(IC) = V(IC) - BH(I,J,1)*YDT(J) - BDTQ(I,J)*Y(J)                0 3955
80  CONTINUE                                                             C 3956
C                                                                           0 3957
      IF (NLAM.GT. 1) GO TO 305                                         0 3958
      V(1) = V(1)/D(1)                                                  C 3959
      GO TO 310                                                         0 3960
C      USE DECOMPOSED BMB AND D ALONG WITH V IN BAKSLV TO SOLVE 11-6  0 3961
C      AND RETURN LAGRANGE MULTIPLIERS IN V ARRAY                     C 3962
305 CALL BAKSLV (BMB,NLAM,V,D,KBMB)                                     0 3963
C                                                                           0 3964
310 LEG = LEND(1)                                                         0 3965
C      FILL OUT EQUATION 11-1 ACCOUNTING FOR ACTION OF CONSTRAINT      0 3966
C      FORCES. THE BM ARRAY WAS OBTAINED IN GETBMB                     0 3967
DO 85 I=1,6                                                                0 3968
      ILN = IV(I,1)                                                      0 3969
      IF (ILN.EQ. 0) GO TO 85                                           0 3970
      F = V(ILN)                                                         0 3971
DO 86 J=1,LEQ                                                             0 3972
86  YDT(J) = YDT(J) + F*BM(1,J,1)                                       C 3973
85  CONTINUE                                                             0 3974
C                                                                           0 3975
DO 90 L=2,NH                                                              0 3976
      NOBQ = ITCPUL(1,L)                                                0 3977
      NOBP = ITOPUL(2,L)                                                0 3978
      LQ = 2*L - 2                                                       0 3979
      LP = LQ + 1                                                        0 3980
      LOG = LCU(NOBQ) - 1                                               C 3981

```

```

      LCP = LCCO(NCBP) - 1
      LEQ = LENO(NCBQ)
      LCP = LENO(NCBP)
      DO 90 I=1,6
      ILN = IV(1,L)
      IF (ILN .EQ. 0) GO TO 90
      F = V(ILN)
      DO 95 J=1,LEQ
      LGCJ = LCO + J
95  YDT(LGCJ) = YDT(LGCJ) + F*DM(I,J,LQ)
      DO 96 J=1,LEP
      LCPJ = LCP + J
96  YDT(LCPJ) = YDT(LCPJ) + F*LM(I,J,LP)
90  CONTINUE
C
      IF (.NOT. LEQ) RETURN
      WRITE(NCT,1003)
1003 FORMAT (' THE YDT ARRAY IS ')
      CALL WRITES (YDT,1,NEQ,1)
      RETURN
      END

```

0 3982
C 3983
0 3984
C 3985
C 3986
0 3987
0 3988
0 3989
0 3990
C 3991
0 3992
C 3993
0 3994
0 3995
0 3996
C 3997
0 3998
C 3999
C 4000
0 4001
0 4002


```

      SUBROUTINE LINEAR (NR,NC)
C
C                                     DEBUG # 40
      IMPLICIT REAL*8 (A-H,O-Z)
C
C SUBROUTINE ESTABLISHES FIRST PARTIAL DERIVATIVES OF A YDOT
C DESCRIBED FUNCTIONAL AT AN INITIAL STATE, Y, USING QUADRATIC
C INTERPOLATION FUNCTIONS.
C   REFER TO VOL 1 SECTION III 'SYNTHESIS AND ANALYSIS OF THE
C   LINEARIZED SYSTEM' FOR THEORETICAL DEVELOPMENT
C
C THE PARTIAL DERIVATIVE MATRIX IS WRITTEN COLUMNWISE ON UNIT NU.
C CALLS SUBROUTINE YDOT
      COMMON /PKWORK/
      *      PR(250,5)
      COMMON /TAPENO/
      *      NTAPE1,NTAPE2,NTAPE3
      COMMON /VECTOR/
      *      Y(250),YDT(250)
      COMMON /VINDEP/
      *      INDEP(250)
      COMMON /LDSIZE/ NX,NY,NOLTA,NXSS,NB,NJO,NY2,NJ2
C
C -----SUBROUTINE ARGUMENT DESCRIPTIONS-----
C NR      = INPUT  NUMBER OF ROWS IN PARTIAL DERIVATIVE MATRIX.
C NC      = INPUT  NUMBER OF COLUMNS IN PARTIAL DERIVATIVE MATRIX.
C FFGM DYN520
C NC = NEQ
C NR = NEQ + NAUX
C
      DIMENSION FY(250,3),Z(250),ZNEW(250),IV(250)
      EQUIVALENCE (PR(1),FY(1)),(PR( 751),Z(1)),(PR(1001),ZNEW(1))
      EQUIVALENCE (INDEP(1),IV(1))
      DATA NCT/ 6 /

```

0 4003
0 4004
C 4005
0 4006
0 4007
0 4008
0 4009
0 4010
0 4011
C 4012
0 4013
0 4014
C 4015
14 4016
0 4017
0 4018
0 4019
20 4020
C 4021
21 4022
0 4023
0 4024
0 4025
0 4026
0 4027
0 4028
0 4029
0 4030
0 4031
78 4032
79 4033
C 4034
0 4035

C	DATA PCCN,PMIN/ 1.0-02,1.0-04 /	0 4036
C	***** CAUTION EPS1 CAN BE PROBLEM SENSITIVE	0 4037
C	DATA EPS1,EPS2/ 1.0-06,1.0-04 /	0 4038
C	COMMON /LDEBUG/ DEBUG(120)	0 4039
	LOGICAL DEBUG,LEQU	0 4040
	EQUIVALENCE (LEQU,DEBUG(40))	0 4041
C	IF (LEQU) WRITE(NOT,1000)	0 4042
1000	FORMAT (' SUBROUTINE LINLAK ')	0 4043
	NU = NTAPE2	0 4044
C	ESTABLISH OUTPUT SIZE OF PARTIAL DERIVATIVE MATRIX	0 4045
C	NJC = 0	0 4046
	NX = 0	0 4047
C	DO 5 I=1,NR	0 4048
C	IV(I)=0 IF COORDINATE I IN STATE VECTOR Y A DEPENDENT COORD.	0 4049
	IF (IV(I) .NE. 0) NJC = NJC+1	0 4050
	IF (I .GT. NC) GO TO 5	0 4051
	IF (IV(I) .NE. 0) NX=NX+1	0 4052
5	CONTINUE	0 4053
C	REWIND NL	0 4054
C	LCAD FY(1) WITH YDT, ETA=0	0 4055
	DO 20 I=1,NR	0 4056
20	FY(I,1) = YDT(I)	0 4057
C	DO 200 L=1,NC	0 4058
	IF (IV(L) .EQ. 0) GO TO 200	0 4059
	IFLG = 0	0 4060
	EPSV = EPS1	0 4061
32	CONTINUE	0 4062
	DY = PCCN * Y(L)	0 4063
	IF (DABS(DY).LT,PMIN) DY=PMIN	0 4064
C	YY = Y(L)	0 4065
	Y(L) = Y(L) + DY	0 4066
	CALL YDGT	0 4067
	Y(L) = YY	0 4068
C	LCAD FY(3) WITH YDT, ETA=1	0 4069
	DO 30 I=1,NR	0 4070
30	FY(I,3) = YDT(I)	0 4071
C	YY = Y(L)	0 4072
	Y(L) = Y(L) + 0.5DU * DY	0 4073
	CALL YDGT	0 4074
	Y(L) = YY	0 4075
C	LCAD FY(2) WITH YDT, ETA=.5	0 4076
	DO 35 I=1,NR	0 4077
35	FY(I,2) = YDT(I)	0 4078
C	ZLRG = 0.000	0 4079
	DO 50 I=1,NR	0 4080
C	EVALUATE EQUATION 111-16	0 4081
		0 4082
		0 4083
		0 4084
		0 4085
		0 4086
		0 4087
		0 4088
		0 4089
		0 4090
		0 4091
		0 4092
		0 4093
		0 4094
		0 4095

E1 = -3.00 * FY(1,1) + 4.00 * FY(1,2) - FY(1,3)	0 4096
C EVALUATE EQUATION 111-17	0 4097
C ALSO LOOK FOR LARGEST Z(1)	0 4098
Z(1) = E1 / DY	0 4099
ZAB = DABS(Z(1))	0 4100
IF(ZAB.LT.EPS1) Z(1) = 0.000	0 4101
IF(ZAB.GT.ZLRG) ZLRG = ZAB	0 4102
50 CONTINUE	0 4103
ITR = 0	0 4104
C	0 4105
60 DY = 0.500*DY	0 4106
C INCREMENT SIZE HAS BEEN CUT IN HALF	0 4107
C	0 4108
C LOAD F(3) WITH YDT, ETA=.5	0 4109
DO 70 I=1,NR	0 4110
70 FY(1,3) = FY(1,2)	0 4111
C	0 4112
YY = Y(L)	0 4113
Y(L) = Y(L) + 0.500 * DY	0 4114
CALL YDCT	0 4115
Y(L) = YY	0 4116
C	0 4117
C LOAD FY(2) WITH YDT, ETA=.25	0 4118
DO 80 I=1,NR	0 4119
80 FY(1,2) = YDT(I)	0 4120
C	0 4121
DO 90 I=1,NR	0 4122
C EVALUATE EQUATION 111-16	0 4123
E1 = -3.00 * FY(1,1) + 4.00 * FY(1,2) - FY(1,3)	0 4124
C EVALUATE EQUATION 111-17	0 4125
ZNEW(1) = E1/DY	0 4126
IF (DABS(ZNEW(1)) .LT. EPS1) ZNEW(1) = 0.00	0 4127
90 CONTINUE	0 4128
C	0 4129
C COMPARE Z AND ZNEW TO SEE IF NEARLY EQUAL	0 4130
IF(.NOT.LEQ0) GO TO 21	0 4131
WRITE (NCT,1002) ITR,L	0 4132
CALL WRITES(Z,1,NR,1)	0 4133
WRITE (NCT,1003) ITR,L	0 4134
CALL WRITES(ZNEW,1,NR,1)	0 4135
1002 FORMAT (' Z ARRAY ITERATION',I3,' FOR COLUMN',I3)	0 4136
1003 FORMAT (' ZNEW ARRAY ITERATION',I3,' FOR COLUMN',I3)	0 4137
21 CONTINUE	0 4138
DO 100 I=1,NR	0 4139
C	0 4140
IF (IV(I) .EQ. 0) GO TO 100	0 4141
C	0 4142
C CONVERGENCE TEST	0 4143
DN = DABS(Z(1))	0 4144
DN1 = DABS(ZNEW(1))	0 4145
C	0 4146
IF (DN1 .GT. DN) DN = DN1	0 4147
IF (DN .LT. EPS1) GO TO 100	0 4148
G1 = DABS(ZNEW(1) - Z(1)) / DN	0 4149
C	0 4150
IF (G1 .LE. EPS2) GO TO 100	0 4151
C	0 4152
ITR = ITR + 1	0 4153
C GIVE UP AFTER 5 TRYS AND 2 EPS1 CHANGES	0 4154
IF(ITR.LE.5) GO TO 90	0 4155

IF(IFLG.GT.0) GO TO 31	0 4156
IFLG = 1	0 4157
EPS1 = 1.0D-06*ZLRG	0 4158
WRITE(NGT,2002) L, EPS1	0 4159
GO TO 32	0 4160
31 IF(IFLG.GT.1) GO TO 33	0 4161
IFLG = 2	0 4162
EPS1 = 1.0D-04*ZLRG	0 4163
WRITE(NGT,2002) L, EPS1	0 4164
GO TO 32	0 4165
33 EPS1 = EPSV	0 4166
GO TO 399	0 4167
96 CONTINUE	0 4168
DO 98 J=1,NR	0 4169
95 Z(J) = ZNEW(J)	0 4170
GO TO 60	0 4171
100 CONTINUE	0 4172
C	0 4173
C COMPLETION OF THE DO 100 LOOP INDICATES WE HAVE ACCEPTED	0 4174
C ZNEW(I) ,I=1,NR.	0 4175
C	0 4176
C NOW PACK PARTIAL DERIVATIVES INTO A NJQ LONG VECTOR.	0 4177
C	0 4178
J=0	0 4179
DO 110 I=1,NR	0 4180
IF (IV(I) .EQ.0) GO TO 110	0 4181
J=J+1	0 4182
ZNEW(J) = ZNEW(I)	0 4183
110 CONTINUE	0 4184
C	0 4185
WRITE (NG) (ZNEW(J),J=1,NJQ)	0 4186
C	0 4187
IF(.NOT.LEQU) GO TO 200	0 4188
WRITE(NGT,1001) L	0 4189
1001 FORMAT (' COLUMN',15,' OF THE PARTIAL DERIVATIVE MATRIX IS')	0 4190
CALL WRITES(ZNEW,1,NJQ,1)	0 4191
C	0 4192
200 CONTINUE	0 4193
RETURN	0 4194
C	0 4195
999 WRITE (NGT,2001) I,L,OY,Z(I),ZNEW(I)	0 4196
2001 FORMAT (1H1,////,20X,	0 4197
* 36HSGURROUTINE LINEAR FAILED TO CONVERGE ,/,20X,	0 4198
* 28HIN 5 ITERATIONS ON ELEMENT ,/,10X,	0 4199
*4H1 = ,13,/,10X,	0 4200
*4HL = ,13,/,10X,19HLAST Y INCREMENT = ,D12.4,/,10X,	0 4201
*7HZ = D12.4,/,10X,	0 4202
*7HZNEW = D12.4)	0 4203
2002 FORMAT (10X,'CONVERGENCE PROBLEMS IN COLUMN',13,' OF LINEAR EPS1 R	0 4204
*ESET TO',D12.4,' FOR THIS COLUMN')	0 4205
C	0 4206
STOP	0 4207
END	0 4208

SUBROUTINE INVIMP (A,R,N,KA)	C 4209
C	
C USED TO INVERT THE MASS MATRIX	DEBUG # 41 0 4210
C IMPLICIT REAL*8(A-H,O-Z)	0 4211
C LOGICAL LDEBUG,LEQU	0 4212
C COMMON /LDEBUG/ LDEBUG(120)	0 4213
C EQUIVALENCE (LEQU,DEBUG(41))	0 4214
C	0 4215
C *****	0 4216
C MATRIX A MUST BE SYMMETRIC. MAX N = XX	0 4217
C *****	0 4218
C	0 4219
C DIMENSION A(KA,1),R(KA,1),CL(22)	0 4220
C	8C 4221
C DATA EPS,NOT / 1.0-06, 0/	0 4222
C	0 4223
C IF (.NOT.LEQU) GO TO 1	0 4224
C WRITE(NCT,1000)	0 4225
C WRITE(NCT,1001)	0 4226
C CALL WRITES(A,N,N,KA)	0 4227
C 1 CONTINUE	0 4228
C L = 1	0 4229
C IF (A(1,1) .LT. EPS) GO TO 999	0 4230
C	0 4231
C R(1,1) = 1.0+00/A(1,1)	0 4232
C DO 100 L = 2,N	0 4233
C N1 = L - 1	0 4234
C DO 10 I=1,N1	0 4235
C CL(I) = 0.0+00	0 4236
C DO 10 J=1,N1	0 4237
C 10 CL(I) = CL(I) + R(1,J)*A(J,L)	0 4238
C S = A(L,L)	0 4239
C DO 20 I=1,N1	0 4240
C 20 S = S - A(I,L)*CL(I)	0 4241
C IF (ABS(S) .LT. EPS) GO TO 999	0 4242
C S = 1.0+00/S	0 4243
C DO 30 I=1,N1	0 4244
C V = -S*CL(I)	0 4245
C R(I,L) = V	0 4246
C R(L,I) = R(I,L)	0 4247
C DO 30 J=1,N1	0 4248
C R(I,J) = R(I,J) - V*CL(J)	0 4249
C 30 R(J,I) = R(I,J)	0 4250
C R(L,L) = S	0 4251
C 100 CONTINUE	0 4252
C IF (.NOT.LEQU) RETURN	0 4253
C WRITE(NCT,1002)	0 4254
C CALL WRITES(R,N,N,KA)	0 4255
C 1000 FORMAT (' SUBROUTINE INVIMP ')	0 4256
C 1001 FORMAT (' MASS MATRIX IS ')	0 4257
C 1002 FORMAT (' INVERSE OF MASS MATRIX IS ')	0 4258
C	0 4259
C RETURN	0 4260
C	0 4261
C 999 WRITE(NCT,900) L	0 4262
C 900 FORMAT(/,5X,32HSINGULAR MATRIX IN INVIMP AT L =,15,9H STOP RUN)	0 4263
C STOP	0 4264
C	0 4265
C END	0 4266
	0 4267

	SUBROUTINE DCCM2 (U,D,N,KR)		0 4268
C		DEBUG # 42	0 4269
	IMPLICIT REAL*8(A-H,O-Z)		0 4270
C	DECOMPOSE U INPUT INTO UPPER TRIANGULAR ELEMENTS OF U AND		0 4271
C	ELEMENTS OF THE DIAGONAL MATRIX D SUCH THAT		0 4272
C	U(INPUT) = U**T*D*U		0 4273
C	DIAGONAL AND LOWER TRIANGULAR ELEMENTS OF OUTPUT U		0 4274
C	CONTAIN GARBAGE		0 4275
C	U SHOULD HAVE ONES ON DIAGONAL AND ZEROS BELOW IT		0 4276
C	NOT DONE HERE TO SAVE TIME, OAKSLV KNJS THIS		0 4277
C	ALGORITHM USED ASSUMES THAT U IS SYMMETRIC		0 4278
	DIMENSION U(KR,1),D(1)		0 4279
	DATA EPS,NOT /1.0-15, 6/		0 4280
C			0 4281
	IF (N .EQ. 1) GO TO 20		0 4282
	NM1 = N - 1		0 4283
	DO 15 L=1,NM1		0 4284
	IF (DABS(U(L,L)) .LT. EPS) GO TO 998		0 4285
	LP1 = L + 1		0 4286
	D(L) = U(L,L)		0 4287
	DO 15 I=LP1,N		0 4288
	S = U(L,I)/U(L,L)		0 4289
	DO 10 J=I,N		0 4290
	10 U(I,J) = U(I,J) - S*U(L,J)		0 4291
	15 U(L,I) = S		0 4292
	20 D(N) = U(N,N)		0 4293
	RETURN		0 4294
C			0 4295
	998 WRITE (NCT,1001)		0 4296
	1001 FORMAT (1H1,40HMATRIX SINGULAR, DCCM2, PROGRAM STOPPED.)		0 4297
	STOP		0 4298
	END		0 4299

	SUBROUTINE WRITES (A,NR,NC,KR)		0 4300
C		DEBUG # 43	0 4301
C	PRINT REAL TWO DIMENSIONAL ARRAY IN FORM A FORMAT		0 4302
	REAL*8 A		0 4303
	DIMENSION A(KR,1),ICHEAD(10)		0 4304
	DATA NOT / 6/		0 4305
	DATA ICHEAD/4H(1),4H(2),4H(3),4H(4),4H(5),		0 4306
	* 4H(6),4H(7),4H(8),4H(9),4H(10) /		0 4307
C			0 4308
	2010 FORMAT (8X,10(7X,A4))		0 4309
	2030 FORMAT (1X,215,2X,1P10D11.3)		0 4310
C			0 4311
	LR = 10		0 4312
	IF (NC .LT. LR) LR = NC		0 4313
	WRITE (NCT,2010) (ICHEAD(L),L=1,LR)		0 4314
	DO 60 I=1,NR		0 4315
	JS = 1		0 4316
	10 JE = JS + 9		0 4317
	IF (JE .GT. NC) JE = NC		0 4318
	WRITE (NCT,2030) I,JS, (A(I,J),J=JS,JE)		0 4319
	IF (JE .EQ. NC) GO TO 60		0 4320
	JS = JS + 10		0 4321

	GO TO 10	0 4322
	60 CONTINUE	0 4323
C		0 4324
	RETURN	0 4325
	END	0 4326
	SUBROUTINE ROTDH	0 4327
C		0 4328
	IMPLICIT REAL*8(A-H,O-Z)	0 4329
C		0 4330
	USED TO COMPUTE ARRAYS RLL AND COL SEE VOL 11,PAGE 18	0 4331
C		0 4332
	INTERMEDIATE ROTATIONS STORED IN RH	0 4333
C		0 4334
	VELOCITY TRANSFORMATION STORED IN PIN	0 4335
	COMMON /BHDSRJ/	0 4336
*	BH(6,18,11),BS(6,18,15),ROL(3,3, 6),DJL(3, 6)	2 4337
	COMMON /HANDS /	0 4338
*	HATH(3,12,10),SIGH(3,12,10),HATS(3,12,15),SLOS(3,12,15)	4 4339
	COMMON /PINRP /	0 4340
*	PIN(3,3, 6), RP2(3,3, 6), RP3(3,3, 6)	13 4341
	COMMON /SPECIF/	0 4342
*	BETAH(6, 6),BETAND(6, 6),AMC(2, 5),RH(3,3,30),RS(3,3,30),	16 4343
*	BH(3,35),DS(3,30),IMO(3, 5),NMDW(6, 6),IFSMW(15),	17 4344
*	NB,NH,NSPT,NCFMU,NDELTA,ITOPOL(2, 6),IRGFLX(6),IHDATA(7, 6),	18 4345
*	LUCC(14),LENU(14),NU,NBETA,NLAM,NEO	19 4346
	COMMON /VECTOR/	0 4347
*	Y(250),YDT(250)	20 4348
C		0 4349
	DIMENSION DF(3),ANGF(3),RDTF(3,3),RW(3,3),	0 4350
*	ITOPW(2, 6),IUP(2, 6),NBODF(6)	81 4351
C		0 4352
	DATA IIST,NOT / 0, 6/	0 4353
	LOGICAL DEBUG,LEQU	0 4354
	COMMON /LDEBUC/ DEBUG(120)	0 4355
	EQUIVALENCE (LEQU,DEBUG(44))	0 4356
C		0 4357
	IF(LEQU) WRITE(NOT,1000)	0 4358
1000	FORMAT (' SUBROUTINE ROTDH ')	0 4359
	IF (IIST.EQ. 1) GO TO 500	0 4360
	IIST = 1	0 4361
	DO 105 I=1,2	0 4362
	DO 105 J=1,NH	0 4363
105	ITCPW(1,J) = ITOPOL(1,J)	0 4364
	IS = 1	0 4365
	DO 106 N=1,NB	0 4366
106	NBODF(N) = 0	0 4367
	ITCPW(1,1) = 0	0 4368
	NOP = 1	0 4369
	ICP(1,1) = 1	0 4370
	IDP(2,1) = 1	0 4371
125	DO 110 I=1,2	0 4372
	DO 110 J=1,NH	0 4373
	IF (ITOPW(1,J).NE. 15) GO TO 110	0 4374
	NOP = NOP + 1	0 4375
	IUP(1,J) = NOP	

IOP(2,J) = ITOPW(1,J)	0 4376
IF (1.EQ. 2) GO TO 115	0 4377
IOP(1,J) = -NOP	0 4378
IOF(2,J) = ITOPW(2,J)	0 4379
115 IS1 = IOP(2,J)	0 4380
NBCDF(IS1) = 1	0 4381
ITCPW(1,J) = 0	0 4382
ITCPW(2,J) = 0	0 4383
116 CONTINUE	0 4384
NBCDF(15) = 0	0 4385
IF (NOP.EQ. NH) GO TO 120	0 4386
DO 116 N=1,NB	0 4387
IF (NBCDF(N).EQ. 0) GO TO 116	0 4388
IS = N	0 4389
GO TO 125	0 4390
116 CONTINUE	0 4391
GO TO 999	0 4392
120 DO 130 J=1,NH	0 4393
JOPA = ICP(1,J)	0 4394
JOP = 1ABS(JOPA)	0 4395
ITCPW(1,JOP) = J*JOPA/JOP	0 4396
130 ITCPW(2,JOP) = IOP(2,J)	0 4397
IF(.NOT.LEQU) GO TO 80	0 4398
WRITE (NCT,1001)	0 4399
1001 FORMAT (' ITOPW ARRAY IS ')	0 4400
CALL WRITIS(ITOPW,2,NH,2)	0 4401
80 CONTINUE	0 4402
C	0 4403
500 CONTINUE	C 4404
C LCAD BETAH WITH UPDATED ANGLES FROM Y STATE VECTOR	C 4405
NBET = LCCU(2*NB + 1) - 1	0 4406
DO 5 J=1,NH	0 4407
DO 5 I=1,6	C 4408
II = I + 1	0 4409
IF (IHDATA(II,J).EQ. 1) GO TO 5	C 4410
NBET = NBET + 1	0 4411
BETAH(I,J) = Y(NBET)	0 4412
5 CONTINUE	0 4413
C	0 4414
DO 10 I=2,NH	0 4415
NOBQ = ITCPOL(1,I)	C 4416
NOBP = ITCPOL(2,I)	0 4417
C LR1 = 1,7,13,19,...	C 4418
C LD1 = 1,8,15,22,...	0 4419
LR1 = 1 + 6*(I-2)	0 4420
LD1 = 1 + 7*(I-2)	0 4421
LR2 = LR1 + 1	0 4422
LR3 = LR1 + 2	0 4423
LR4 = LR1 + 3	0 4424
LR5 = LR1 + 4	0 4425
LR6 = LR1 + 5	0 4426
LD2 = LD1 + 1	0 4427
LD3 = LD1 + 2	0 4428
LD4 = LD1 + 3	0 4429
LD5 = LD1 + 4	0 4430
LD6 = LD1 + 5	0 4431
LD7 = LD1 + 6	0 4432
NMQ = IRGFLX(NOBQ)	0 4433
NMP = IRGFLX(NOBP)	0 4434
C	C 4435

```

C      GO TO 15 IF BODY CONTAINING Q FRAME RIGID      0 4436
C      IF (NMQ .EQ. 0) GO TO 15                      0 4437
C                                                    0 4438
C      ACCOUNT FOR BODY FLEXIBILITY TO GET POSITION DH(LD3) AND      0 4439
C      ORIENTATION RH(LR3) OF Q FRAME AT HINGE 1 WRT ITS BODY FRAME 0 4440
C      LU = LUCO(NBDO+NB)                             0 4441
C      LHS = 2*I - 3                                   0 4442
C      CALL MULT3 (HATH(1,1,LHS),Y(LU),DF,3,NMQ,1,3,1,1) 0 4443
C      CALL MULT3 (SIGH(1,1,LHS),Y(LU),ANGF,3,NMQ,1,3,1,1) 0 4444
C      CALL ROTTR (3,1,ANGF,ROTF,DUM,DUM)              0 4445
C      CALL MULT3 (ROTF,RH(1,1,LR1),RH(1,1,LR3),3,3,3,3,3,3) 0 4446
C      DO 12 J=1,3                                     0 4447
C      12 DH(J,LD3) = DH(J,LD1) + DF(J)                0 4448
C                                                    0 4449
C      GO TO 20 IF BODY CONTAINING P FRAME RIGID      0 4450
C      15 IF (NMP .EQ. 0) GO TO 20                    0 4451
C                                                    0 4452
C      ACCOUNT FOR BODY FLEXIBILITY TO GET POSITION DH(LD4) AND      0 4453
C      ORIENTATION RH(LR4) OF P FRAME AT HINGE 1 WRT ITS BODY FRAME 0 4454
C      LU = LUCO(NBPP+NB)                             0 4455
C      LHS = 2*I - 2                                   0 4456
C      CALL MULT3 (HATH(1,1,LHS),Y(LU),DF,3,NMP,1,3,1,1) 0 4457
C      CALL MULT3 (SIGH(1,1,LHS),Y(LU),ANGF,3,NMP,1,3,1,1) 0 4458
C      CALL ROTTR (3,1,ANGF,ROTF,DUM,DUM)              0 4459
C      CALL MULT3 (ROTF,RH(1,1,LR2),RH(1,1,LR4),3,3,3,3,3,3) 0 4460
C      DO 17 J=1,3                                     0 4461
C      17 DH(J,LD4) = DH(J,LD2) + DF(J)                0 4462
C                                                    0 4463
C      20 DO 25 J=1,3                                  0 4464
C      JP3 = J + 3                                     0 4465
C      ANGF(J) = BETAH(J,1)                           0 4466
C      25 DH(J,LD5) = BETAH(JP3,1)                    0 4467
C      IT = IHDATA(1,1)                               0 4468
C                                                    0 4469
C      ACCOUNT FOR RELATIVE MOTION BETWEEN P AND Q FRAMES AT HINGE 1 0 4470
C      RH(LR5) USE INPUTTED BETAH ANGLES              0 4471
C      CALL ROTTR (3,IT,ANGF,RH(1,1,LR5),DUM,DUM)      0 4472
C                                                    0 4473
C      GET FI INVERSE MATRIX OF EQUATION 11-19 VOL 1 0 4474
C      PIN(1) = ROTATION TRANSFORMATION MATRIX RELATING BODY      0 4475
C      FIXED ANGULAR RATES TO EULER RATES BETWEEN P AND      0 4476
C      Q FRAME AT HINGE 1(NON-ORTHONORMAL)            0 4477
C      CALL ROTTR (1,IT,ANGF,PIN(1,1,1),RP2(1,1,1),RP3(1,1,1)) 0 4478
C                                                    0 4479
C      DO 35 L=1,3                                     0 4480
C      DO 35 J=1,3                                     0 4481
C      35 RW(L,J) = RH(J,L,LR3)                        0 4482
C                                                    0 4483
C      RH(LR6) = RH(LR4)*RH(LR5)*RH(LR3)**IT          0 4484
C      TRANSFORMATION BETWEEN BODY FIXED FRAMES OF BODIES      0 4485
C      CONTIGUOUS AT HINGE 1                             0 4486
C      CALL MULT3 (RH(1,1,LR4),RH(1,1,LR5),ROTF,3,3,3,3,3,3) 0 4487
C      CALL MULT3 (ROTF,RW,RH(1,1,LR6),3,3,3,3,3,3,3) 0 4488
C                                                    0 4489
C      DH(LD6) = VECTOR BETWEEN BODY FIXED REFERENCE FRAMES OF      0 4490
C      BODIES CONTIGUOUS AT HINGE 1                    0 4491
C      CALL MULT3 (RH(1,1,LR4),DH(1,LD5),DF,3,3,1,3,3,1) 0 4492
C      CALL MULT3 (RH(1,1,LR6),DH(1,LD3),ANGF,3,3,1,3,3,1) 0 4493
C      DO 40 J=1,3                                     0 4494
C      40 DH(J,LD6) = DH(J,LD4) + DF(J) - ANGF(J)      0 4495

```


C		0 4496
C	10 CONTINUE	0 4497
C	ALL RELATIVE POSITION AND ORIENTATION DATA OF P-Q FRAME AT	0 4498
C	EACH HINGE POINT COMPUTED AND STORED IN OH AND RH ARRAYS	0 4499
C		0 4500
C		0 4501
C		0 4502
C	PROCEED TO COMPUTE:	0 4503
C	ROL(K) ROTATION TRANSFORMATION MATRIX BODY K FIXED AXIS	0 4504
C	TO INERTIAL AXIS	0 4505
C	DOL(K) POSITION VECTOR FROM INERTIAL ORIGIN TO BODY	0 4506
C	K FIXED REFERENCE	0 4507
	DO 45 J=1,3	0 4508
	JP3 = J + 3	0 4509
	DOL(J,1) = BETAH(JP3,1)	0 4510
45	ANGF(J) = BETAH(J,1)	0 4511
	IT = IHCATA(1,1)	0 4512
	CALL ROTTR (3,IT,ANGF,ROL(1,1,1),DUM,DUM)	0 4513
	CALL ROTTR (1,IT,ANGF,PIN(1,1,1),RP2(1,1,1),RP3(1,1,1))	0 4514
C		0 4515
	DO 50 J=2,NH	0 4516
	NOH = ITOPW(1,J)	0 4517
	LR0J = ITOPW(2,J)	0 4518
	IF (NOH .LT. 0) GO TO 52	0 4519
	LR6 = 6*(NOH - 1)	0 4520
	LR0 = ITCPOL(2,NOH)	0 4521
	CALL MULT3 (ROL(1,1,LR0),RH(1,1,LR6),ROL(1,1,LR0J),3,3,3,3,3)	0 4522
	GO TO 50	0 4523
52	NOH = -NOH	0 4524
	LR6 = 6*(NOH - 1)	0 4525
	LR0 = ITCPOL(1,NOH)	0 4526
	DO 53 I=1,3	0 4527
	DO 53 L=1,3	0 4528
53	RW(1,L) = RH(L,1,LR6)	0 4529
	CALL MULT3 (ROL(1,1,LR0),RW,ROL(1,1,LR0J),3,3,3,3,3)	0 4530
50	CONTINUE	0 4531
C		0 4532
	DO 60 J=2,NH	0 4533
	LR0 = ITCPOL(2,J)	0 4534
	LD6 = 7*(J-1) - 1	0 4535
	LD7 = LD6 + 1	0 4536
60	CALL MULT3 (ROL(1,1,LR0),DH(1,LD6),DH(1,LD7),3,3,1,3,3,3)	0 4537
C		0 4538
	DO 70 J=2,NH	0 4539
	NOH = ITOPW(1,J)	0 4540
	LD0J = ITOPW(2,J)	0 4541
	IF (NOH .LT. 0) GO TO 72	0 4542
	LD7 = 7*(NOH - 1)	0 4543
	LD0 = ITCPOL(2,NOH)	0 4544
	DO 74 I=1,3	0 4545
74	DOL(1,LD0J) = DOL(1,LD0) + DH(1,LD7)	0 4546
	GO TO 70	0 4547
72	NOH = - NOH	0 4548
	LD7 = 7*(NOH - 1)	0 4549
	LD0 = ITCPOL(1,NOH)	0 4550
	DO 73 I=1,3	0 4551
73	DOL(1,LD0J) = DOL(1,LD0) - DH(1,LD7)	0 4552
70	CONTINUE	0 4553
C		0 4554
	IF (.NOT.LEQU) RETURN	0 4555

IF (LEQU) WRITE (NOT,1000)	0 4610
1000 FORMAT (' SUBROUTINE BHGENR ')	0 4611
C	0 4612
IF (I1ST .EQ. 1) GO TO 100	0 4613
I1ST = 1	0 4614
LR = 2*NHMAX - 1	0 4615
JR = 6 + NMUBOD	0 4616
DO 5 L=1,LR	0 4617
DO 5 I=1,6	0 4618
DO 5 J=1,JR	0 4619
5 EH(I,J,L) = ZRO	0 4620
C	0 4621
100 DO 10 I=1,3	0 4622
IP3 = 1 + 3	0 4623
DO 10 J=1,3	0 4624
JP3 = J + 3	0 4625
EH(I,J,1) = PIN(I,J,1)	0 4626
10 EH(IP3,JP3,1) = ROL(I,J,1)	0 4627
C	0 4628
DO 20 L=2,NH	0 4629
LQ = 2*L - 2	0 4630
LP = LQ + 1	0 4631
C	0 4632
C	0 4633
LQ = 2,4,6,...	0 4634
C	0 4635
LP = 3,5,7,...	0 4636
C	0 4637
BH(LP) = EQUATION 11-82, VOL 1	0 4638
C	0 4639
BH(LQ) = EQUATION 11-83, VOL 1	0 4640
C	0 4641
LR3 = 6*(L-2) + 3	0 4642
LR4 = LR3 + 1	0 4643
LR5 = LR3 + 2	0 4644
LD3 = 7*(L-2) + 3	0 4645
LD4 = LD3 + 1	0 4646
DO 25 I=1,3	0 4647
DO 25 J=1,3	0 4648
W1(J,I) = RH(I,J,LR3)	0 4649
W2(J,I) = RH(I,J,LR5)	0 4650
25 EH(J+3,I+3,LP) = -RH(I,J,LR4)	0 4651
CALL MULT3 (RH(1,1,LR3),W1,BH(4,4,LQ),3,3,3,3,0)	0 4652
CALL MULT3 (PIN(1,1,L),W1,BH(1,1,LQ),3,3,3,3,0)	0 4653
CALL MULT3 (W2,BH(4,4,LP),W1,3,3,3,3,6,3)	0 4654
CALL MULT3 (PIN(1,1,L),W1,BH(1,1,LP),3,3,3,3,3,6)	0 4655
CALL SKEWV3 (DH(1,LD3),W1,3,3)	0 4656
CALL SKEWV3 (DH(1,LD4),W2,3,3)	0 4657
CALL MULT3 (BH(4,4,LQ),W1,BH(4,1,LQ),3,3,3,6,3,6)	0 4658
CALL MULT3 (DH(4,4,LP),W2,BH(4,1,LP),3,3,3,6,3,6)	0 4659
NOBQ = ITOPOL(1,L)	0 4660
NOBP = ITOPOL(2,L)	0 4661
NMG = IRGFLX(NOBQ)	0 4662
NMP = IRGFLX(NOBP)	0 4663
C	0 4664
IS Q-FRAME IN A RIGID BODY?	0 4665
C	0 4666
IF (NMQ .EQ. 0) GO TO 30	0 4667
LHS = 2*L - 3	0 4668
CALL MULT3 (BH(1,1,LQ),SIGN(1,1,LHS),BH(1,7,LQ),3,3,NMQ,6,3,6)	0 4669
CALL MULT3 (BH(4,4,LQ),HATH(1,1,LHS),BH(4,7,LQ),3,3,NMQ,6,3,6)	0 4670
C	0 4671
IS P-FRAME IN A RIGID BODY?	0 4672
C	0 4673
30 IF (NMP .EQ. 0) GO TO 20	0 4674
LHS = 2*L - 2	0 4675

CALL MULT3 (BH(1,1,L),SIGH(1,1,LHS),BH(1,7,L),J,J,NMP,6,3,6)	0 4670
CALL MULT3 (BH(4,4,L),HATH(1,1,LHS),BH(1,7,L),J,J,NMP,6,3,6)	0 4671
20 CONTINUE	0 4672
C	0 4673
IF(.NOT.LEQU) RETURN	0 4674
LX = 2*N1-1	0 4675
LMY = 6+NM000	0 4676
DO 40 L=1,LMX	0 4677
WRITE (NCT,1001) L	0 4678
1001 FORMAT(' BH(' ,12,') IS ')	0 4679
40 CALL WRITES(BH(1,1,L),6,LMY,6)	0 4680
RETURN	0 4681
END	0 4682
SUBROUTINE ROTDS	0 4683
C	0 4684
DEBUG # 46	0 4685
C	0 4686
COMPUTE RS ROTATION TRANSFORMATION BETWEEN SENSOR AXIS SYSTEM	0 4687
C AND BODY FIXED AXIS SYSTEM VOL II,PAGE 19	0 4688
C NOTHING DONE IF SENSOR CN RIGID BODY	0 4689
IMPLICIT REAL*8(A-H,O-Z)	0 4690
C	0 4691
COMMON /HANDS /	0 4692
* HATH(3,12,10),SIGH(3,12,10),HATS(3,12,15),SIGS(3,12,15)	4 4693
COMMON /SPECIF/	0 4694
* BETAH(6, 6),BETAHD(6, 6),AMD(2, 5),RH(3,3,30),RS(3,3,30),	16 4695
* DH(3,35),DS(3,30),IMU(3, 5),NMOW(6, 3),IFSMW(15),	17 4696
* NB,NH,NSPT,NCFMO,NDELTA,ITCPOL(2, 6),IRGFLX(6),IHDATA(7, 6),	18 4697
* LUCL(14),LENU(14),NU,NBETA,NLAM,NEQ	19 4698
COMMON /VECTOR/	0 4699
* Y(250),YDT(250)	20 4700
C	0 4701
DATA NOT/6 /	0 4702
LOGICAL DEBUG,LEQU	0 4703
COMMON /LDEBUG/ DEBUG(120)	0 4704
EQUIVALENCE (LEQU,DEBUG(40))	0 4705
DIMENSION DF(3),AF(3),RF(3,3)	0 4706
C	0 4707
IF(LEQU) WRITE(NOT,1000)	0 4708
DO 10 L=1,NSPT	0 4709
N0B = IFSMW(L)	0 4710
IF (IRGFLX(N0B) .EQ. 0) GO TO 10	0 4711
LR1 = 2*L - 1	0 4712
LR2 = LR1 + 1	0 4713
LO = LUCL(NB+N0B)	0 4714
LE = LENU(NB+N0B)	0 4715
CALL MULT3 (HATS(1,1,L),Y(LO),DF,3,LE,1,3,1,1)	0 4716
CALL MULT3 (SIGS(1,1,L),Y(LO),AF,3,LE,1,3,1,1)	0 4717
CALL ROTTR (3,1,AF,RF,DUM,DUM)	0 4718
CALL MULT3 (RF,RS(1,1,LR1),RS(1,1,LR2),3,3,3,3,3,3)	0 4719
DO 15 I=1,3	0 4720
15 DS(I,LR2) = DS(I,LR1) + DF(I)	0 4721
C	0 4722
10 CONTINUE	0 4723
IF(.NOT.LEQU) RETURN	0 4723

	11 = 2*NSPT	0 4724
	WRITE(NOT,1002) (1,(RS(J,L,I),J=1,3),L=1,3),I=1,11)	0 4725
	WRITE(NCT,1003) (1,(DS(J,I),J=1,3),I=1,11)	0 4726
1000	FORMAT (' SUBROUTINE RCTDS ENTERED')	0 4727
1002	FORMAT (1X,' RS(',12,') =',1P9D11.3)	0 4728
1003	FORMAT (1X,' DS(',12,') =',1P3D11.3)	0 4729
C		0 4730
	RETURN	0 4731
	END	0 4732
	SUBROUTINE BSGENR	0 4733
C		0 4734
C	DEBUG # 47	0 4735
C	GET KINEMATICAL COEFFICIENTS FOR ALL SENSOR POINTS	0 4736
C	SEE VOL 11,PAGE 17	0 4737
C	EQUATIONS ANALOGOUS TO THOSE PROGRAMMED IN BSGENR	0 4738
	IMPLICIT REAL*8(A-H,O-Z)	0 4739
C		0 4740
	COMMON /BHBSRD/	0 4741
*	BH(6,18,11),BS(6,18,15),ROL(3,3, 6),JOL(3, 6)	2 4742
	COMMON /HANDS /	0 4743
*	HATH(3,12,10),SIGH(3,12,10),HATS(3,12,15),SIGS(3,12,15)	4 4744
	COMMON /MAXMUM/	0 4745
*	NBMAX,NHMAX,NSPMAX,NMwMAX,NMWBOB,NMDBOB,KMU,KY,KU	0 4746
	COMMON /NUMBRS/	0 4747
*	ZRO,ONE,TWO,THREE	0 4748
	COMMON /SPECIF/	0 4749
*	BETAH(6, 6),BETAHD(6, 6),AMO(2, 5),RH(3,3,30),RS(3,3,30),	16 4750
*	DR(3,35),DS(3,30),IMG(3, 5),NMDW(6, 5),IFTSMW(15),	17 4751
*	NB,NH,NSPT,NUFMQ,NDELTA,ITOPOL(2, 6),IRGFLX(6),IHDATA(7, 6),	18 4752
*	LOCU(14),LENU(14),NU,NBETA,NLAM,NEQ	19 4753
C		0 4754
	DIMENSION W(3,3)	0 4755
C		0 4756
	DATA I1ST / 0 /	0 4757
	DATA NOT/6/	0 4758
	LOGICAL DEBUG,LEQU	0 4759
	COMMON /LDEBUG/ DEBUG(120)	0 4760
	EQUIVALENCE (LEQU,DEBUG(47))	0 4761
C		0 4762
	IF(LEQU) WRITE(NOT,1000)	0 4763
1000	FORMAT (' SUBROUTINE BSGENR')	0 4764
	IF (I1ST .EQ. 1) GO TO 20	0 4765
C		0 4766
	JR = 6 + NMDBOB	0 4767
	DO 5 L=1,NSPT	0 4768
	DO 5 I=1,6	0 4769
	DO 5 J=1,JR	0 4770
	5 BS(1,J,L) = ZRO	0 4771
C		0 4772
20	DO 10 L=1,NSPT	0 4773
	NOB = IFTSMW(L)	0 4774
	LE = IRGFLX(NOB)	0 4775
	IF (LE .EQ. 0 .AND. I1ST .EQ. 1) GO TO 10	0 4776
	LR2 = 2*L	0 4777

DO 15 I=1,3	0 4778
IP3 = I + 3	0 4779
DO 15 J=1,3	0 4780
JP3 = J + 3	0 4781
ES(J,1,L) = RS(1,J,LR2)	0 4782
15 ES(JF3,IP3,L) = RS(1,J,LR2)	0 4783
CALL SKEWV3 (DS(1,LR2),W,3,3)	0 4784
CALL MULT3 (BS(1,1,L),W,BS(4,1,L),3,3,3,6,3,6)	0 4785
IF (LE.EQ. 0) GO TO 10	0 4786
CALL MULT3 (BS(1,1,L),SIGS(1,1,L),BS(1,7,L),3,3,LE,6,3,6)	0 4787
CALL MULT3 (BS(1,1,L),HATS(1,1,L),DS(4,7,L),3,3,LE,6,3,6)	0 4788
10 CONTINUE	0 4789
C	0 4790
11ST = 1	0 4791
IF(.NOT.LEQU) RETURN	0 4792
LMY = 0+NMDBOD	0 4793
DO 25 L=1,NSPT	0 4794
WRITE (NOT,1001)L	0 4795
1001 FORMAT (' BS(' ,I2,') IS '	0 4796
25 CALL WRITES(DS(1,1,L),0,LMY,6)	0 4797
RETURN	0 4798
END	0 4799
SUBROUTINE GETBMB	0 4800
C	0 4801
IMPLICIT REAL*8(A-H,O-Z)	0 4802
C	0 4803
COMMON /AMDBW /	0 4804
* AMU(22,22, 6),BW(36,113)	1 4805
COMMON /BHDSRD/	0 4806
* BH(6,18,11),BS(6,18,15),ROL(3,3, 6),JUL(3, 6)	2 4807
COMMON /IVCONS/	0 4808
* IV(6, 6)	9 4809
COMMON /MAXMOM/	0 4810
* NBMAX,NHMAX,NJPMAX,NMWMAX,NMWBCD,NMDBOD,K4U,KY,KU	0 4811
COMMON /NUMGRS/	0 4812
* ZRC,ONE,TWO,THRE	0 4813
COMMON /SPECIF/	0 4814
* BETAH(6, 6),BETAHD(6, 6),AMC(2, 5),RH(3,3,30),RS(3,3,30),	16 4815
* DH(3,35),DS(3,30),IMU(3, 5),NMOW(6, 6),IFISM(15),	17 4816
* NB,NH,NSPT,NDFMD,NDELTA,ITCPOL(2, 6),IRGFLX(6),IHDATA(7, 6),	18 4817
* LUCL(14),LENU(14),NU,NBETA,NLAM,NEQ	19 4818
C	0 4819
DIMENSION BMB(36,36),BM(6,22,11), ITOP(6, 6),*R(18)	82 4820
EQUIVALENCE (BW(1),BMB(1)), (BW(1297),BM(1))	77 4821
DATA NOT/6 /	0 4822
LOGICAL DEBUG,LEQU	0 4823
COMMON /LDEBUG/ DEBUG(120)	0 4824
EQUIVALENCE (LEQU,DEBUG(48))	0 4825
C	0 4826
DATA 11ST / 0 /	0 4827
:	0 4828
IF(LEQU) WRITE(NOT,1000)	0 4829
1000 FORMAT (' SUBROUTINE GETBMB')	0 4830
IF (11ST.EQ. 1) GO TO 100	0 4831

11ST = 1	0 4832
DO 5 I=1,NH	0 4833
DO 5 J=1,NB	0 4834
5 ITCP(I,J) = 0	0 4835
ITCP(1,1) = 1	0 4836
DO 3 I=1,6	0 4837
DO 3 J=1,NH	0 4838
3 IV(I,J) = 0	0 4839
C	0 4840
C LOAD IV ARRAY, USED TO SIGNAL CONSTRAINT CONDITION	0 4841
IC = 0	0 4842
DO 7 J=1,NH	0 4843
DO 7 I=1,6	0 4844
IP1 = I + 1	0 4845
IF (IHDATA(IP1,J) .EQ. 0) GO TO 7	0 4846
IC = IC + 1	0 4847
IV(I,J) = IC	0 4848
7 CONTINUE	0 4849
C	0 4850
C ITCP ARRAY USED TO FIND STORAGE LOCATION IN BH ARRAY FOR	0 4851
C PARTICULAR VELOCITY TRANSFORMATIONS, EQUATIONS 11-82,83 VOL 1	0 4852
C ITOP(L,NQ) = STORAGE LOCATION FOR Q FRAME AT HINGE L BODY NQ	0 4853
C ITOP(L,NP) = STORAGE LOCATION FOR P FRAME AT HINGE L BODY NP	0 4854
DO 10 L=2,NH	0 4855
LQ = 2*L - 2	0 4856
LP = LQ + 1	0 4857
NQ = ITCPOL(1,L)	0 4858
NP = ITCPOL(2,L)	0 4859
ITOP(L,NQ) = LQ	0 4860
10 ITOP(L,NP) = LP	0 4861
C	0 4862
C SELECTIVE ROW MULTIPLICATIONS USED TO COMPUTE	0 4863
C BMB = B * M**(-1) * B**T	0 4864
C THIS MATRIX USED IN EQUATION 11-6 TO FIND LAGRANGE MULTIPLIERS	0 4865
C SEE PAGE 37, VOL 1	0 4866
CC SAVE EM = B * M**(-1) NEEDED FOR EQUATION 11-6 EVALUATION	0 4867
100 LEQ = 0 + IRGFLX(1)	0 4868
LMQ = LEND(1)	0 4869
CALL MLTSR (BH,AMU,BM,LEQ,LMQ,1,IV,KMU)	0 4870
C	0 4871
DO 20 L=2,NH	0 4872
NOBQ = ITCPOL(1,L)	0 4873
NOBP = ITCPOL(2,L)	0 4874
LEG = IRGFLX(NOBBQ) + 0	0 4875
LEP = IRGFLX(NOBBP) + 0	0 4876
LMQ = LEND(NOBBQ)	0 4877
LMP = LEND(NOBBP)	0 4878
LQ = 2*L - 2	0 4879
LP = LQ + 1	0 4880
CALL MLTSR (BH(1,1,LQ),AMU(1,1,NOBBQ),BM(1,1,LQ),LEQ,LMQ,L,IV,KMU)	0 4881
CALL MLTSR (BH(1,1,LP),AMU(1,1,NOBBP),BM(1,1,LP),LEP,LMP,L,IV,KMU)	0 4882
20 CONTINUE	0 4883
C	0 4884
DO 25 I=1,NLAM	0 4885
DO 25 J=1,NLAM	0 4886
25 GME(I,J) = ZRO	0 4887
C	0 4888
DO 30 N=1,NB	0 4889
LE = IRGFLX(N) + 6	0 4890
DO 35 L=1,NH	0 4891

```

      IF (ITOP(L,N) .EQ. 0) GO TO 35      0 4892
      DO 40 I=1,NH                        0 4893
      IF (ITOP(I,N) .EQ. 0) GO TO 40      0 4894
      LBM = ITOP(L,N)                     0 4895
      LBT = ITOP(I,N)                     0 4896
      DO 50 M=1,6                          0 4897
      ML = IV(M,L)                         0 4898
      IF (ML .EQ. 0) GO TO 50              0 4899
      DO 55 K=1,LE                          0 4900
55  WR(K) = EM(M,K,LBM)                   0 4901
      DO 60 J=1,6                          0 4902
      JI = IV(J,I)                         0 4903
      IF (JI .LT. ML) GO TO 60             0 4904
      S = ZRO                               0 4905
      DO 65 K=1,LE                          0 4906
65  S = S + WR(K)*BH(J,K,LBT)             0 4907
      EMB(ML,JI) = BMB(ML,JI) + S         0 4908
      GO CONTINUE                          0 4909
50  CONTINUE                              0 4910
40  CONTINUE                              0 4911
35  CONTINUE                              0 4912
30  CONTINUE                              0 4913
C                                          0 4914
      IF (.NOT. LEGU) RETURN                0 4915
      WRITE (NCT,1001)                     0 4916
1001 FORMAT (' THE BMD ARRAY IS ')         0 4917
      L09 = 0*NHMAX                        0 4918
      CALL WRITES(NMB,NLAM,NLAM,L09)       0 4919
      RETURN                               0 4920
      END                                  0 4921

      SUBROUTINE MGEN                      0 4922
C                                          0 4923
C                                          0 4924
C          GENERATE DEFORMATION DEPENDENT MASS MATRIX CONTAINED IN      0 4925
C          EQUATION 11-86 VOL 1 VIA EQUATIONS 11-87,88,89                0 4926
C          IMPLICIT REAL*8(A-H,O-Z)      0 4927
C                                          0 4928
C          COMMON /AMUBW /              0 4929
*      AMU(22,22, 6),BW(36,113)        1 4930
C          COMMON /BHBSRD/              0 4931
*      BH(6,18,11),BS(6,18,13),ROL(3,3, 6),DOL(3, 6) 2 4932
C          COMMON /GSSAVE/              0 4933
*      GSS(12,9, 6)                    3 4934
C          COMMON /INTGRL/              0 4935
*      A4(171, 6),ACUF(9,12, 6),BCCF(6,12, 6),      5 4936
*      CUF11(12,12, 6),CUF22(12,12, 6),CUF33(12,12, 6),AK(12,12, 6), 6 4937
*      CUF12(12,12, 6),CUF13(12,12, 6),CUF23(12,12, 6),AD(12,12, 6), 7 4938
*      COFX1(12,12, 6),COFX2(12,12, 6),COFYZ(12,12, 6) 8 4939
C          COMMON /MAXMUM/              0 4940
*      NBMAX,NHMAX,NSPMAX,NMWMAX,NMWBOD,NMDBOD,KMU,KY,KU 0 4941
C          COMMON /NUMBRS/              0 4942
*      ZRO,ONE,TWO,TRES                 0 4943
C          COMMON /SPECIF/              0 4944
*      BETAH(6, 6),BETAHD(6, 6),AMD(2, 5),RH(3,3,30),RS(3,3,30), 16 4945

```



```

*      CH(3,35),DS(3,30),IMU(3, 5),NMCW(6, 6),IFISM(15),      17 4946
*      NB,NH,NSPT,NQFMU,NDELTA,ITCPOL(2, 6),IRGFLX( 6),IHDATA(7, 6), 18 4947
*      LUCC(14),LENU(14),NU,NBETA,NLAM,NEQ      19 4948
      COMMON /VECTOR/
*      Y(250),YDT(250)      20 4950
C      0 4951
      DIMENSION RW(3,12),CW(12,3),VW(9),WV(6)      83 4952
      DATA NOT/ 6 /      0 4953
      LOGICAL DEBUG,LEQU      0 4954
      COMMON /LDEBUG/ DEBUG(120)      0 4955
      EQUIVALENCE (LEQU,DEBUG(49))      0 4956
      IF (LEQU) WRITE(NOT,1000)      0 4957
1000 FORMAT (' SUBROUTINE MGEN')      0 4958
C      0 4959
      KM = NMDEUD      0 4960
      DO 10 N=1,NB      0 4961
      LD = LCCU(NB+N)      0 4962
      LE = LENU(NB+N)      0 4963
      KNT = 0      0 4964
      NP6 = 6 + LE      0 4965
      DO 12 I=1,NP6      0 4966
      DO 12 J=1,NP6      0 4967
      KNT = KNT + 1      0 4968
12  AMU(I,J,N) = AM(KNT,N)      0 4969
      IF (LE .EQ. 0) GO TO 50      0 4970
      CALL MULT3 (BCOF(1,1,N),Y(LD),VW,6,LE,1,6,1,1)      0 4971
      CALL MULT3 (COF11(1,1,N),Y(LD),CW(1,1),LE,LE,1,KM,1,KM)      0 4972
      CALL MULT3 (COF22(1,1,N),Y(LD),CW(1,2),LE,LE,1,KM,1,KM)      0 4973
      CALL MULT3 (COF33(1,1,N),Y(LD),CW(1,3),LE,LE,1,KM,1,KM)      0 4974
      CALL MULT3 (Y(LD),COF12(1,1,N),RW(1,1),1,LE,LE,1,KM,3)      0 4975
      CALL MULT3 (Y(LD),COF13(1,1,N),RW(2,1),1,LE,LE,1,KM,3)      0 4976
      CALL MULT3 (Y(LD),COF23(1,1,N),RW(3,1),1,LE,LE,1,KM,3)      0 4977
C      0 4978
C      0 4979
C      PEEL OFF DATA FOR GRAVITY GRADIENT EFFECTS ON ELASTIC COORDINATES      0 4980
C      0 4981
      DO 8 J=1,LE      0 4982
      GGS(J,1,N) = CW(J,1)      0 4983
      GGS(J,2,N) = CW(J,2)      0 4984
      GGS(J,3,N) = CW(J,3)      0 4985
      GGS(J,7,N) = RW(1,J)      0 4986
      GGS(J,8,N) = RW(2,J)      0 4987
8  GGS(J,9,N) = RW(3,J)      0 4988
C      0 4989
      CALL MULT3 (Y(LD),CW,WV(1),1,LE,3,1,KM,1)      0 4990
      CALL MULT3 (RW,Y(LD),WV(4),3,LE,1,3,1,1)      0 4991
      DO 15 I=1,3      0 4992
15  AMU(I,1,N) = AMU(I,1,N) + TWO*VW(I) + WV(I)      0 4993
      AMU(I,2,N) = AMU(I,2,N) - VW(4) - WV(4)      0 4994
      AMU(I,3,N) = AMU(I,3,N) - VW(5) - WV(5)      0 4995
      AMU(I,3,N) = AMU(I,3,N) - VW(6) - WV(6)      0 4996
      CALL MULT3 (ACOF(1,1,N),Y(LD),VW,9,LE,1,9,1,1)      0 4997
      DO 17 J=1,3      0 4998
      JP3 = J + 3      0 4999
      DO 17 I=1,3      0 5000
      IJ = J + 3*(I-1)      0 5001
17  AMU(I,JP3,N) = AMU(I,JP3,N) + VW(IJ)      0 5002
      CALL MULTAD (Y(LD),COFYZ(1,1,N),AMU(1,7,N),1,LE,LE,1,KM,KMU)      0 5003
      CALL MULTAD (Y(LD),COFXZ(1,1,N),AMU(2,7,N),1,LE,LE,1,KM,KMU)      0 5004
      CALL MULTAD (Y(LD),COFXY(1,1,N),AMU(3,7,N),1,LE,LE,1,KM,KMU)      0 5005
C      0 5005

```

CALL MULT3 (CDF12(1,1,N),Y(L0),CW(1,1),LE,LE,1,KM,1,KM)	C 5006
CALL MULT3 (CDF13(1,1,N),Y(L0),CW(1,2),LE,LE,1,KM,1,KM)	O 5007
CALL MULT3 (CDF23(1,1,N),Y(L0),CW(1,3),LE,LE,1,KM,1,KM)	O 5008
C	C 5009
C FINISH FEELING OFF GRAVITY GRADIENT DATA	O 5010
C	O 5011
DO 26 J=1,LE	C 5012
GGG(J,4,N) = CW(J,1)	O 5013
GGG(J,5,N) = CW(J,2)	O 5014
26 GGG(J,6,N) = CW(J,3)	O 5015
C	O 5016
C	O 5017
C ACCOUNT FOR ALL ACTIVE WHEELS IN BODY N	O 5018
50 NMWVS = NMGW(2,N)	O 5019
IF (NMWVS .EQ. 0) GO TO 110	O 5020
NMW = NMGW(1,N)	O 5021
LEBS = 6 + LE	O 5022
NV = 0	O 5023
J1 = LEBS + 1	O 5024
J2 = LEBS + NMWVS	O 5025
DO 70 L=1,NMW	O 5026
LP2 = L + 2	O 5027
NOMW = NMOW(LP2,N)	O 5028
IF (IMU(3,NOMW) .EQ. 0) GO TO 70	O 5029
C SEE VCL I,PAGE 47,GET ALL BUT 1,1 TERM IN EQ 11-109	O 5030
C	O 5031
NV = NV + 1	O 5032
LV = 6 + LE + NV	O 5033
NOSP = IMU(1,NOMW)	O 5034
NA = IMU(2,NOMW)	O 5035
AJS = AMU(2,NOMW)	O 5036
DO 75 J=1,LEBS	O 5037
75 AMU(J,LV,N) = AJS*BS(NA,J,NOSP)	O 5038
DO 76 J=J1,J2	O 5039
76 AMU(LV,J,N) = ZRC	O 5040
AMU(LV,LV,N) = AJS	O 5041
70 CONTINUE	C 5042
C	O 5043
110 LEU = LENU(N)	O 5044
DO 77 I=1,LEU	O 5045
DO 77 J=1,LEU	O 5046
77 AMU(J,I,N) = AMU(I,J,N)	O 5047
C	O 5048
IF(.NOT.LEQU) GO TO 20	O 5049
WRITE (NCT,1001) N	O 5050
1001 FORMAT (' AMU, MASS MATRIX FOR BODY',I3)	O 5051
LOB = 6+NMDBUD+NMWBUD	O 5052
CALL WRITES(AMU(1,1,N),LEU,LEU,LOB)	O 5053
IF(LE.EQ.0) GO TO 20	O 5054
WRITE (NCT,1002) N	O 5055
1002 FORMAT (' GGS, GRAVITY GRADIENT EFFECTS ON MODAL COORDINATES OF BC	O 5056
*CY',I3)	C 5057
CALL WRITES(GGS(1,1,N),LE,9,NMCEUD)	O 5058
20 CONTINUE	O 5059
10 CONTINUE	O 5060
C	O 5061
RETURN	C 5062
END	O 5063

```

SUERCUTINE GRVGRD (GGV)                                0 5064
C                                                         0 5065
IMPLICIT REAL*8(A-H,O-Z)                                0 5066
DIMENSION GGV(1)                                         0 5067
C                                                         0 5068
COMMON /AMUBW /                                           0 5069
*   AMU(22,22, 6),UBW(36,113)                            1 5070
COMMON /BHBSRD/                                           0 5071
*   BH(6,18,11),BS(6,18,15),ROL(3,3, 6),DOL(3, 6)       2 5072
COMMON /GGDATA/                                           0 5073
*   GAMGI(3),GMAG,RCMAG                                    0 5074
COMMON /GGSAVE/                                           0 5075
*   GGS(12,9, 6)                                           3 5076
COMMON /INTGRL/                                           0 5077
*   AM(171, 6),ACCF(9,12, 6),BCOF(6,12, 6),             5 5078
*   COF11(12,12, 6),COF22(12,12, 6),COF33(12,12, 6),AK(12,12, 6), 6 5079
*   COF12(12,12, 6),COF13(12,12, 6),COF23(12,12, 6),AD(12,12, 6), 7 5080
*   COFXY(12,12, 6),COFXZ(12,12, 6),COFYZ(12,12, 6)     8 5081
COMMON /MAXMUM/                                           0 5082
*   NDMAX,NHMAX,NSPMAX,NM*MAX,NMWEDD,NMDSDD,KMU,KY,KU    0 5083
COMMON /NUMBRS/                                           0 5084
*   ZRC,ONE,TWO,THRS                                       0 5085
COMMON /SPECIF/                                           0 5086
*   BETAH(6, 6),DETAMD(6, 6),AMO(2, 5),RH(3,3,30),RS(3,3,30), 16 5087
*   DH(3,35),DS(3,30),IMO(3, 5),NMOW(6, 6),IFISM*(15), 17 5088
*   NB,NH,NSPT,NUFMD,NDELTA,ITCPOL(2, 6),IRGFLX( 6),IHDATA(7, 6), 18 5089
*   LUCU(14),LENU(14),NU,NBETA,NLAM,NEO                  19 5090
COMMON /VECTOR/                                           0 5091
*   Y(250),YDT(250)                                       20 5092
C                                                         0 5093
DIMENSION GAMGL(3),V(3)                                   0 5094
DATA NOT / 6/                                             0 5095
LOGICAL DEBUG,LEQU                                       0 5096
COMMON /LDEBUG/ DEBUG(120)                               0 5097
EQUIVALENCE (LEQU,DEBUG(50))                             0 5098
IF (LEQU) WRITE(NOT,1000)                                0 5099
1000 FORMAT(' SUBROUTINE GRVGRD ')                        0 5100
C   **** CAUTION WHEN APPLYING GRAVITY GRADIENT CAPABILITY 0 5101
C   ***** IT IS NOT APPLICABLE TO ALL PROBLEMS         0 5102
C   ***** UPDATE UNDER DEVELOPEMENT, CALL BUDLEY      0 5103
C   FROM DYN510:                                           0 5104
C   GAMGI - UNIT VECTOR IN DIRECTION OF GRAVITY FIELD INERTIAL 0 5105
C   COORDINATES                                           0 5106
C   GMAG - LOCAL GRAVITATIONAL ACCELERATION               0 5107
C   RCMAG - RADIOS VECTOR TO VICINITY OF SPACECRAFT       0 5108
C   FROM GRAVITY SOURCE                                   0 5109
IF (GMAG.EQ. ZRO) RETURN                                  0 5110
IF (RCMAG.LE. ONE) GO TO 999                              0 5111
DO 10 N=1,NB                                              0 5112
LOU = LDCU(N)                                             0 5113
LO = LDCU(N+NB)                                           0 5114
LE = LENU(N+NB)                                           0 5115
C                                                         0 5116
C   EVALUATE EQUATION 11-125 FIRST THREE ELEMENTS OF THE 0 5117
C   CONTRIBUTION TO THE FORCE VECTOR DUE TO GRAVITY GRADIENT 0 5118
C   GAMGL - UNIT GRAVITY DIRECTION VECTOR LOCAL COORDINATES 0 5119
CALL MULT3 (GAMGI,ROL(1,1,N),GAMGL,1,3,1,3,1)          0 5120
C   AMU(4,1,N) STARTING LOCATION FOR FIRST MASS MOMENT COEFFICIENTS 0 5121
CALL MULT3 (AMU(4,1,N),GAMGL,GGV(LOU),3,3,1,<40,1,1)   0 5122
CALL MULT3 (AMU(1,1,N),GAMGL,V,3,3,1,KMU,1,1)          0 5123

```

```

GGV(LCU ) = GMAG*(GGV(LCU )
* + TRFS*(GAMGL(2)*V(3) - GAMGL(3)*V(2))/RCMAG)
GGV(LCU+1) = GMAG*(GGV(LCU+1)
* + TRFS*(GAMGL(3)*V(1) - GAMGL(1)*V(3))/RCMAG)
GGV(LCU+2) = GMAG*(GGV(LCU+2)
* + TRFS*(GAMGL(1)*V(2) - GAMGL(2)*V(1))/RCMAG)
C
C      GET SECOND THREE TERMS DEFINED BY EQUATION 11-12b
C      PICK OUT MASS MOMENT COEFFICIENTS
V(1) = AMU(3,5,N)
V(2) = AMU(1,5,N)
V(3) = AMU(2,4,N)
S = ZRU
GCM = -GMAG*AMU(4,4,N)
GCR = -GMAG/RCMAG
LUP2 = LCU + 2
DO 20 I=1,3
GGV(LUP2+I) = GCM*GAMGL(I) + GCR*V(I)
20 S = S + GAMGL(I)*V(I)
S = -TRFS*S*GCR
DO 25 I=1,3
25 GGV(LUP2+I) = GGV(LUP2+I) + S*GAMGL(I)
C
C      GET THE FORCE DUE TO GRAVITY ACTING ON THE DEFORMATION MODES
C      USE EQUATION 11-12f
IF (LE .EQ. 0) GO TO 10
LOP6 = LCU + 6
V(1) = GAMGL(1)*RCMAG
V(2) = GAMGL(2)*RCMAG
V(3) = GAMGL(3)*RCMAG
CALL MULTS (V,AMU(4,7,N),GGV(LOP6),1,3,LE,1,KMU,1)
CALL MULTAD (AMU(7,7,N),Y(LD),GGV(LOP6),LE,LE,1,KMU,1,1)
LOP5 = LCU + 5
DO 30 J=1,LE
GGV(LOP5+J) = GCR*GGV(LOP5+J)
* + GCR*(BCOF(1,J,N) + BCOF(2,J,N) + BCOF(3,J,N))/TWO
30 CONTINUE
C
C      V(1) = CNE - TWO*GAMGL(1)*GAMGL(1)
V(1) = CNE - TWO*GAMGL(1)*GAMGL(1)
V(2) = CNE - TWO*GAMGL(2)*GAMGL(2)
V(3) = CNE - TWO*GAMGL(3)*GAMGL(3)
S = TRFS*GMAG/(TWO*RCMAG)
C
DO 40 J=1,LE
GGV(LOP5+J) = GGV(LOP5+J)
* + S*(V(1)*(BCOF(1,J,N) + GGS(J,1,N))
* + V(2)*(BCOF(2,J,N) + GGS(J,2,N))
* + V(3)*(BCOF(3,J,N) + GGS(J,3,N))
* + TWO*GAMGL(1)*GAMGL(2)*(BCOF(4,J,N) + GGS(J,4,N) + GGS(J,7,N))
* + TWO*GAMGL(1)*GAMGL(3)*(BCOF(5,J,N) + GGS(J,5,N) + GGS(J,8,N))
* + TWO*GAMGL(2)*GAMGL(3)*(BCOF(6,J,N) + GGS(J,6,N) + GGS(J,9,N))
40 CONTINUE
C
IF(.NOT.LEQU) GO TO 10
WRITE(NT,1001) N
1001 FORMAT (' CONTRIBUTION TO FORCE VECTOR FROM GRAVITY GRADIENT EFFEC
*1S BCDY',13)
J = LOP5+LE
CALL WRITES(GGV,1,J,1)
10 CONTINUE

```

RETURN	0 5184
C	0 5185
999 WRITE (NCT,2001)	0 5186
2001 FORMAT (1H1,25HRCMAG = 0., SUBROUTINE GRVGRD)	0 5187
STOP	0 5188
END	0 5189
SUBROUTINE BDOTOP (L,BDTQ,EDTP)	0 5190
C	0 5191
IMPLICIT REAL*8(A-H,O-Z)	0 5192
C	0 5193
GET LOWER CASE B DOT MATRIX CALLED FOR IN EQ. 11-6	0 5194
C	0 5195
REFER TO APPENDIX C VOL 1 FOR THEORY ALSO PAGES 36-38	0 5196
C	0 5197
EXIT WITH THE B DOT MATRIX TO BE USED IN EQ. 11-6 ASSOCIATED	0 5198
C	0 5199
WITH BODY L	0 5200
C	0 5201
DIMENSION BDTQ(6,1),BDTP(6,1)	0 5202
C	0 5203
COMMON /BHUSRO/	0 5204
* BH(6,18,11),BS(6,18,15),ROL(3,3, 6),JL(3, 6)	0 5205
COMMON /HANDS /	0 5206
* HATH(3,12,10),SIGH(3,12,10),HATS(3,12,15),SIGS(3,12,15)	0 5207
COMMON /NUMBRS/	0 5208
* ZRO,ONE,TWO,THRE	0 5209
COMMON /PINKP /	0 5210
* PIN(3,3, 6), RP2(3,3, 6), RP3(3,3, 6)	0 5211
COMMON /SPECIF/	0 5212
* BETAH(6, 6),BETAHD(6, 6),AMO(2, 5),RH(3,3,30),RS(3,3,30),	0 5213
* OH(3,35),DS(3,30),IMU(3, 5),NMOW(6, 6),IFTSMW(15),	0 5214
* NB,NH,NSPT,NQFMD,NDELTA,ITGPOL(2, 6),IRGFLX(6),IHDATA(7, 6),	0 5215
* LUCL(14),LENU(14),NU,NBETA,NLAM,NEQ	0 5216
COMMON /VLCUR/	0 5217
* Y(250),YDT(250)	0 5218
C	0 5219
DIMENSION PINDT(3,3),HXDGN(3),SXDQN(3),HXDPM(3),SXDPM(3),RQP(3,3),	0 5220
* RPM(3,3),RQN(3,3),RPN(3,3),RQM(3,3),DRPJ(3,3),DRQP(3,3),	0 5221
* DRGN(3,3),DRQM(3,3),SNG(3,3),SMP(3,3),WQN(3),WPM(3),WPQP(3),	0 5222
* WSK(3,3), VEC(3)	0 5223
COMMON /MAXMUM/	0 5224
* NBMAX,NHMAX,NSPMAX,NMWMAX,NMWBOC,NMDBOD,KYJ,KY,KU	0 5225
DATA NUT/6 /	0 5226
LOGICAL DEBUG,LEQU	0 5227
COMMON /LDEBUG/ DEBUG(120)	0 5228
EQUIVALENCE (LEQU,DEBUG(51))	0 5229
IF (LEQU) WRITE(NOT,1000)	0 5230
1000 FORMAT (' SUBROUTINE BDOTOP ')	0 5231
C	0 5232
DO 3 I=1,6	0 5233
DO 3 J=1,6	0 5234
BDTQ(I,J) = ZRO	0 5235
3 EDTP(I,J) = ZRO	0 5236
IF (L .EQ. 1) GO TO 100	0 5237
C	0 5238
CC GET O/CT(PI(INVERSE))	0 5239
E2CT = BETAHD(2,L)	0 5240

```

E3CT = BETAMD(3,L)                                0 5238
DO 5 I=1,3                                          0 5239
DO 5 J=1,3                                          0 5240
5 FINDT(1,J) = B2DT*RP2(1,J,L) + B3DT*RP3(1,J,L) 0 5241
C                                                    0 5242
NOBQ = ITOPOL(1,L)                                C 5243
NOBP = ITOPOL(2,L)                                0 5244
LQO = LQCU(NOBQ) + 6                              0 5245
LOP = LQCU(NOBP) + 6                              0 5246
C      THE Q-FRAME IS IN BODY NOBQ IT HAS LEQ FLEXIBLE BODY MODES 0 5247
C      THE P-FRAME IS IN BODY NOBP IT HAS LEP FLEXIBLE BODY MODES 0 5248
LEG = IRGFLX(NOBQ)                                0 5249
LEP = IRGFLX(NOBP)                                0 5250
LHSQ = 2*L - 3                                     0 5251
LHSP = LHSQ + 1                                    0 5252
IF (LEQ .EQ. 0) GO TO 10                          0 5253
CALL MULT3 (HATH(1,1,LHSQ),Y(LQO),HXDQN,3,LEQ,1,3,1,1) C 5254
CALL MULT3 (SIGH(1,1,LHSQ),Y(LQO),SXDNQ,3,LEQ,1,3,1,1) 0 5255
10 IF (LEP .EQ. 0) GO TO 20                        0 5256
CALL MULT3 (HATH(1,1,LHSP),Y(LOP),HXDPM,3,LEP,1,3,1,1) 0 5257
CALL MULT3 (SIGH(1,1,LHSP),Y(LOP),SXDPM,3,LEP,1,3,1,1) 0 5258
20 LRNQ = 6*(L-2) + 3                              0 5259
LRMP = LRNQ + 1                                    C 5260
LRFQ = LRNQ + 2                                    0 5261
LDNQ = 7*(L-2) + 3                                 0 5262
LDMP = LDNQ + 1                                    0 5263
DO 15 I=1,3                                         0 5264
DO 15 J=1,3                                         C 5265
RQP(1,J) = RH(J,1,LRPQ)                            0 5266
RQN(1,J) = RH(J,1,LRNQ)                            0 5267
15 RPM(1,J) = RH(J,1,LRMP)                          C 5268
CALL MULT3 (RQP,RPM,RQM,3,3,3,3,3,3,3)             0 5269
CALL MULT3 (RH(1,1,LRPQ),RCN,RFN,3,3,3,3,3,3,3)    C 5270
CALL SKEWVS (DH(1,LDNQ),SNG,3,3)                   0 5271
CALL SKEWVS (DH(1,LDMP),SMP,3,3)                     0 5272
C                                                    0 5273
IQ = LQCU(NOBQ) - 1                                0 5274
IP = LQCU(NOBP) - 1                                0 5275
DO 25 I=1,3                                          0 5276
WQN(I) = -Y(IQ+I)                                   0 5277
25 WPM(I) = Y(IP+I)                                  0 5278
IF (LEQ .EQ. 0) GO TO 26                          0 5279
DO 27 I=1,3                                          0 5280
27 WQN(I) = WQN(I) - SXDNQ(I)                       0 5281
26 IF (LEP .EQ. 0) GO TO 28                          0 5282
DO 29 I=1,3                                          0 5283
29 WPM(I) = WPM(I) + SXDPM(I)                       0 5284
28 CALL MULT3 (RPM,WPM,WQP,3,3,1,3,1,1,1)          C 5285
CALL MULTAD (RPN,WQN,WQPF,3,3,1,3,1,1,1)           0 5286
CALL SKEWVS (WQPF,WSK,1,3)                          C 5287
CALL MULT3 (WSK,RH(1,1,LRPQ),DRPQ,3,3,3,3,3,3,3)  C 5288
DO 30 I=1,3                                          0 5289
DO 30 J=1,3                                          0 5290
30 DRCP(I,J) = DRPQ(J,I)                            0 5291
C                                                    0 5292
CALL MULT3 (DRPQ,RCN,EDTQ(4,4),3,3,3,3,3,3,6)      C 5293
CALL MULT3 (DRPQ,RPM,DRQM,3,3,3,3,3,3,3)           0 5294
CALL MULT3 (PINDT,RQN,EDTG,3,3,3,3,3,3,6)          0 5295
IF (LEQ .EQ. 0) GO TO 35                            C 5296
CALL MULT3 (RQN,SXDQN,VEC,3,3,1,3,1,1,1)           0 5297

```

CALL SKEWV3 (VEC,WSK,1,3)	0 5298
CALL MULT3 (WSK,RQN,DRQN,3,3,3,3,3,3)	0 5299
CALL MULTAD (RH(1,1,LRPQ),DRQN,BDTQ(4,4),3,3,3,3,3,6)	0 5300
CALL MULTAD (PIN(1,1,L),DRQN,BDTQ,3,3,3,3,3,6)	0 5301
CALL MULT3 (BDTQ(4,4),SNQ,BDTQ(4,1),3,3,3,3,3,6)	0 5302
CALL SKEWV3 (HXDQN,WSK,1,3)	0 5303
CALL MULTAD (RPN,WSK,BDTQ(4,1),3,3,3,3,3,6)	0 5304
CALL MULT3 (BDTQ,SIGH(1,1,LHSQ),BDTQ(1,7),3,3,LEQ,3,3,6)	0 5305
CALL MULT3 (BDTQ(4,4),HATH(1,1,LHSQ),BDTQ(4,7),3,3,LEQ,3,3,6)	0 5306
GO TO 40	0 5307
35 CALL MULT3 (BDTQ(4,4),SNQ,BDTQ(4,1),3,3,3,3,3,6)	0 5308
40 CONTINUE	0 5309
C	0 5310
IF (LEP .EQ. 0) GO TO 50	0 5311
CALL MULT3 (RPM,SDPM,VEC,3,3,1,3,1,1)	0 5312
CALL SKEWV3 (VEC,WSK,1,3)	0 5313
CALL MULT3 (WSK,RPM,BDTP(4,4),3,3,3,3,3,6)	0 5314
CALL MULT3 (PINDT,RQM,BDTP(1,1),3,3,3,3,3,6)	0 5315
CALL MULT3 (BDTP(4,4),SMP,BDTP(4,1),3,3,3,3,3,6)	0 5316
CALL MULTAD (ROP,BDTP(4,4),DRQM,3,3,3,3,3,6)	0 5317
CALL MULTAD (PIN(1,1,L),DRQM,BDTP,3,3,3,3,3,6)	0 5318
CALL SKEWV3 (HXDPM,WSK,1,3)	0 5319
CALL MULTAD (RPM,WSK,BDTP(4,1),3,3,3,3,3,6)	0 5320
DO 55 I=1,3	0 5321
IP3 = I + 3	0 5322
DO 55 J=1,3	0 5323
JP3 = J + 3	0 5324
BDTP(I,J) = -BDTP(I,J)	0 5325
EDTP(IP3,J) = -BDTP(IP3,J)	0 5326
55 EDTP(IP3,JP3) = -BDTP(IP3,JP3)	0 5327
CALL MULT3 (BDTP(1,1),SIGH(1,1,LHSP),BDTP(1,7),3,3,LEP,3,3,6)	0 5328
CALL MULT3 (BDTP(4,4),HATH(1,1,LHSP),BDTP(4,7),3,3,LEP,3,3,6)	0 5329
GO TO 60	0 5330
50 CALL MULT3 (PINDT,RQM,BDTP,3,3,3,3,3,6)	0 5331
CALL MULTAD (PIN(1,1,L),DRQM,BDTP,3,3,3,3,3,6)	0 5332
DO 58 I=1,3	0 5333
DO 58 J=1,3	0 5334
58 BDTP(I,J) = -BDTP(I,J)	0 5335
60 CONTINUE	0 5336
IF (.NOT. LEQ) RETURN	0 5337
WRITE(NOT,1001) L	0 5338
1001 FORMAT (' BDTP ARRAY HINGE',I3)	0 5339
L11 = 6 + LEP	0 5340
CALL WRITES(BDTP,6,L11,6)	0 5341
WRITE(NOT,1002) L	0 5342
1002 FORMAT (' BDTQ ARRAY HINGE',I3)	0 5343
L11 = 6 + LEQ	0 5344
CALL WRITES(BDTQ,6,L11,6)	0 5345
RETURN	0 5346
C	0 5347
100 DO 70 I=1,3	0 5348
70 WQN(I) = -Y(I)	0 5349
CALL SKEWV3 (WQN,WSK,1,3)	0 5350
CALL MULT3 (ROL,WSK,BDTQ(4,4),3,3,3,3,3,6)	0 5351
E2DT = BETAHD(2,1)	0 5352
E3DT = BETAHD(3,1)	0 5353
DO 75 I=1,3	0 5354
DO 75 J=1,3	0 5355
75 BDTQ(I,J) = B2DT*RP2(I,J,1) + B3DT*RP3(I,J,1)	0 5356
C	0 5357

IF(.NOT.LEQU) RETURN	0 5358
WRITE(NCT,1002) L	0 5359
L11 = 0	0 5360
CALL WRITES(JDTQ,6,L11,6)	0 5361
RETURN	0 5362
END	0 5363
SUBROUTINE BAKSLV (BW,NLAM,V,D,KBW)	0 5364
C	DEBUG # 52 0 5365
C	0 5366
C USE DECOMPOSED BMB AND V TO COMPUTE LAGRANGE MULTIPLIERS	0 5367
IMPLICIT REAL*8(A-H,O-Z)	0 5368
DIMENSION BW(KBW,1),V(1),D(1)	0 5369
DATA NCT/6 /	0 5370
LOGICAL DEBUG,LEQU	0 5371
COMMON /LDEBUG/ DEBUG(120)	0 5372
EQUIVALENCE (LEQU,DEBUG(52))	0 5373
C	0 5374
DO 25 I=2,NLAM	0 5375
IM1 = I - 1	0 5376
DO 25 J=1,IM1	0 5377
25 V(I) = V(I) - BW(J,I)*V(J)	0 5378
DO 27 I=1,NLAM	0 5379
27 V(I) = V(I)/D(I)	0 5380
NL1 = NLAM - 1	0 5381
DO 30 I=1,NL1	0 5382
L = NLAM - I	0 5383
LP1 = L + 1	0 5384
DO 30 J=LP1,NLAM	0 5385
30 V(L) = V(L) - BW(L,J)*V(J)	0 5386
C	0 5387
IF(.NOT.LEQU) RETURN	0 5388
WRITE(NCT,1000)	0 5389
1000 FORMAT(' LAGRANGE MULTIPLIERS FROM BAKSLV ARE ')	0 5390
CALL WRITES(V,1,NLAM,1)	0 5391
RETURN	0 5392
END	0 5393
SUBROUTINE SKEWV3(V,SKV,KV,KSKV)	0 5394
C	DEBUG # 53 0 5395
C	0 5396
C	0 5397
C	0 5398
C	0 5399
C	0 5400
C	0 5401
C	0 5402
C	0 5403
C	0 5404
C	0 5405

	SKV(3,1) = V(2,1)	0 5406
	SKV(1,2) = V(3,1)	0 5407
	SKV(3,2) = -SKV(2,3)	0 5408
	SKV(1,3) = -SKV(3,1)	0 5409
	SKV(2,1) = -SKV(1,2)	0 5410
	SKV(1,1) = 0.0 0	0 5411
	SKV(2,2) = 0.0 0	0 5412
	SKV(3,3) = 0.0 0	0 5413
C		0 5414
	RETURN	0 5415
	END	0 5416

	SUBROUTINE MULT3(A,B,Z,NRA,NRB,NCB,KRA,KRB,KRZ)	0 5417
		0 5418
C	DEBUG # 54	0 5419
C		0 5420
C	GENERAL MATRIX PRODUCT OF RECTANGULAR MATRICES	0 5421
C	A(NRA X NRB) * B(NRB X NCB) = Z(NRA X NCB)	0 5422
C		0 5423
	IMPLICIT REAL*8(A-H,O-Z)	0 5424
	DIMENSION A(KRA,1),B(KRB,1),Z(KRZ,1),WR(100)	84 5425
C		0 5426
	DO 20 I=1,NRA	0 5427
	DO 15 J=1,NRB	0 5428
15	WR(J) = A(I,J)	0 5429
	DO 20 J=1,NCB	0 5430
	S = 0.0 0	0 5431
	DO 30 K=1,NRB	0 5432
30	S = S + WR(K)*B(K,J)	0 5433
20	Z(I,J) = S	0 5434
C		0 5435
	RETURN	0 5436
	END	

	SUBROUTINE MULTAD (A,B,Z,NRA,NRB,NCB,KRA,KRB,KRZ)	0 5437
		0 5438
C	MULTIPLY	0 5439
C	A(NRA,NRB) * B(NRB,NCB)	0 5440
C	ADD THE RESULT TO	0 5441
C	Z(NRA,NRB)	0 5442
C	RETURN	0 5443
C		0 5444
	Z = Z + A * B	0 5445
	IMPLICIT REAL*8(A-H,O-Z)	85 5446
	DIMENSION A(KRA,1),B(KRB,1),Z(KRZ,1),WR(100)	0 5447
C		0 5448
	DO 20 I=1,NRA	0 5449
	DO 15 J=1,NRB	0 5450
15	WR(J) = A(I,J)	0 5451
	DO 20 J=1,NCB	0 5452
	S = 0.0 0	0 5453
	DO 30 K=1,NRB	

30	S = S + RW(K)*B(K,J)	0	5454
20	Z(I,J) = Z(I,J) + S	0	5455
C		0	5456
	RETURN	0	5457
	END	0	5458
	SUBROUTINE MLTSR (A,B,C,LE,LM,L,IV,KMU)	0	5459
C		0	5460
	USED TO PERFORM MATRIX	0	5461
C	C = A * B	0	5462
C	WHERE	0	5463
C	C(6,LM) = A(6,LE) * B(LE,LM)	0	5464
C	FOR SELECTED ROWS OF C. SELECTION DEFINED BY IV INTEGER ARRAY	0	5465
	IMPLICIT REAL*8(A-H,O-Z)	0	5466
	DIMENSION A(6,1),B(KMU,1),C(6,1),IV(6,1),RW(18)	86	5467
C		0	5468
	DO 10 I=1,6	0	5469
	IL = IV(I,L)	0	5470
	IF (IL .EQ. 0) GO TO 10	0	5471
	DO 15 K=1,LE	0	5472
15	RW(K) = A(I,K)	0	5473
	DO 20 J=1,LM	0	5474
	S = C.O 0	0	5475
	DO 25 K=1,LE	0	5476
25	S = S + RW(K)*B(K,J)	0	5477
20	C(I,J) = S	0	5478
10	CONTINUE	0	5479
C		0	5480
	RETURN	0	5481
	END	0	5482
	SUBROUTINE TORQUE (G)	0	5483
C		0	5484
	IMPLICIT REAL*8(A-H,O-Z)	0	5485
	DIMENSION G(1)	0	5486
C		0	5487
	COMMON /BHESRD/	0	5488
*	BH(6,18,11),BS(6,18,15),ROL(3,3,6),JOL(3,6)	2	5489
	COMMON /GGSAVE/	0	5490
*	GGS(12,9,6)	3	5491
	COMMON /INTGRL/	0	5492
*	AM(171,6),ACCF(9,12,6),BCOF(6,12,6),	5	5493
*	COF11(12,12,6),COF22(12,12,6),COF33(12,12,6),AK(12,12,6),	6	5494
*	COF12(12,12,6),COF13(12,12,6),COF23(12,12,6),AD(12,12,6),	7	5495
*	COFXY(12,12,6),COFXZ(12,12,6),COFYZ(12,12,6)	8	5496
	COMMON /MAXMUM/	0	5497
*	NBMAX,NHMAX,NSPMAX,NMWMAX,NMWBCD,NMOSUD,KMU,KY,KU	0	5498
	COMMON /MUMENG/	0	5499
*	F(113),PMCM(36),HTGT(3),TCTL(3),ENGKE(6),ENGPE(6),	11	5500
*	TUTKL,TUTPE,TUTLNG,AHICT,ATGTL	0	5501

```

COMMON /NUMBRS/
* ZRO,ONE,TWO,THREE
COMMON /SPECIF/
* BETAH(6, 6),BETAHD(6, 6),AMO(2, 5),RH(3,3,30),RS(3,3,30),
* OH(3,35),DS(3,30),IMD(3, 5),NMOW(6, 6),IFTSMW(15),
* NU,NH,NSPT,NGFMO,NDELTA,ITOPOL(2, 6),IRGFLX( 6),IHDATA(7, 6),
* LUCU(14),LENU(14),NU,NBETA,NLAN,NEQ
COMMON /VECTOR/
* Y(250),YDT(250)
C
C DIMENSION CW(12,3),RW(3,12),V1(12),WSK(3,3),TSHFT( 5),
* TEX(6,15),ISPN(15),V(6),V2(12)
DATA NOT/6/
LOGICAL DEBUG,LEQU
COMMON /LDEBUG/ DEBUG(120)
EQUIVALENCE (LEQU,DEBUG(57))
IF (LEQU) WRITE (NOT,1000)
1000 FORMAT (' SUBROUTINE TORQUE ')
C
CCC SUBROUTINE CONTRL ESTABLISHES THE D/DT(DELTA) USER SUPPLIED --
CALL CONTRL
C
CCC SUBROUTINE EXTOR ESTABLISHES ALL EXTERNAL TORQUES, INCLUDING RCS
CCCC CONTROL TORQUES, ETC., USER SUPPLIED --
CALL EXTOR (TEX,ISPN,NTEX)
IF (NTEX .EQ. 0) GO TO 5
C
DO 65 L=1,NTEX
NSP = ISPN(L)
NBCD = IFTSMW(NSP)
LON = LUCU(NBCD) - 1
LEN = IRGFLX(NBCD) + 6
DO 66 I=1,6
TQ = TEX(I,L)
IF (TQ .EQ. ZRO) GO TO 66
DO 67 J=1,LEN
JL = J + LON
67 G(JL) = G(JL) + TQ*BS(1,J,NSP)
66 CONTINUE
65 CONTINUE
C
CCC SUBROUTINE SHAFTT ESTABLISHES SHAFT TORQUE FOR EACH
CCCC MOMENTUM WHEEL (ZLRUS IT OUT IF CONSTANT SPEED) USER SUPPLIED --
E CALL SHAFTT (TSHFT)
C
CCCC SETUP HINGE SPRING AND DASHPOT RESTORING TORQUES
CCCC ALSO ACCOUNT FOR POTENTIAL ENERGY DUE TO HINGE SPRINGS
CALL KHINGE (G)
C
IF (.NOT.LEQU) GO TO 80
WRITE (NOT,1001)
1001 FORMAT (' G VECTOR ALL FORCES AND TORQUES EXTERNAL TO BODIES ')
CALL WRITES(G,1,NU,1)
80 CONTINUE
C
KM = NMDEUD
C
C CYCLE THROUGH ALL BODIES COMPUTING GYROSCOPIC FORCES AND
C TORQUES ACTING ON THEM
DO 50 N=1,NU

```

```

      LOU = LDCU(N)                                0 5562
      LO = LDCU(N+NB)                              0 5563
      LE = LLEN(N+NB)                              0 5564
      LEU = LE + 6                                  0 5565
      IF (LE .EQ. 0) LEU = 3                        0 5566
      LOU1 = LOU + 1                               0 5567
      LOU2 = LOU + 2                               0 5568
      LOU3 = LOU + 3                               0 5569
      LOU4 = LOU + 4                               0 5570
      LOU5 = LOU + 5                               0 5571
C
C      EVALUATE EQUATION II-58 ADD IT TO THE LOAD VECTOR G 0 5572
C
      CALL SKEWV3 (Y(LOU),WSK,1,3)                 0 5573
      CALL MULTAD (WSK,P(LOU),G(LOU),3,3,1,3,1,1) 0 5574
      CALL MULTAD (WSK,P(LOU3),G(LOU3),3,3,1,3,1,1) 0 5575
      CALL SKEWV3 (Y(LOU3),WSK,1,3)                C 5576
      CALL MULTAD (WSK,P(LOU3),G(LOU),3,3,1,3,1,1) 0 5577
      IF (LE .EQ. 0) GO TO 100                      0 5578
C
C      ADD IN THE MODAL STIFFNESS AND DAMPING CONTRIBUTIONS 0 5579
C      GMISC USED TO ADD IN MISCELLANEOUS EFFECTS SUCH AS 0 5580
C      THERMALLY INDUCED MOTION                      0 5581
      CALL GMISC (N,LE,LO,V2)                      0 5582
      CALL MULT3 (AK(1,1,N),V2,V1,LE,LE,1,KM,1,1) 0 5583
      CALL MULTAD (AD(1,1,N),YDT(LO),V1,LE,LE,1,KM,1,1) 0 5584
      DO 10 J=1,LE                                  0 5585
      I = LOU5 + J                                  0 5586
      10 G(I) = G(I) - V1(J)                        0 5587
C
C      GCS COMPUTED IN MGEN USED HERE IN THE EVALUATION OF 0 5588
C      EQUATION II-91,92,93 AS WELL AS IN THE GRAVITY GRADIENT STUFF 0 5589
C
      DO 15 J=1,LE                                  0 5590
      CW(J,1) = -TWO*Y(LOU)*(BCOF(1,J,N) + GGS(J,1,N)) 0 5591
      * + Y(LOU1)*(BCOF(4,J,N) + GGS(J,4,N) + GGS(J,7,N)) 0 5592
      * + Y(LOU2)*(BCOF(5,J,N) + GGS(J,5,N) + GGS(J,8,N)) 0 5593
      * -Y(LOU3)*ACOF(1,J,N) - Y(LOU4)*ACOF(2,J,N) - Y(LOU5)*ACOF(3,J,N) 0 5594
      CW(J,2) = -TWO*Y(LOU1)*(BCOF(2,J,N) + GGS(J,2,N)) 0 5595
      * + Y(LOU2)*(BCOF(6,J,N) + GGS(J,6,N) + GGS(J,9,N)) 0 5596
      * + Y(LOU3)*(BCOF(4,J,N) + GGS(J,4,N) + GGS(J,7,N)) 0 5597
      * -Y(LOU5)*ACOF(4,J,N) - Y(LOU4)*ACOF(5,J,N) - Y(LOU5)*ACOF(6,J,N) 0 5598
      CW(J,3) = -TWO*Y(LOU2)*(BCOF(3,J,N) + GGS(J,3,N)) 0 5599
      * + Y(LOU3)*(BCOF(5,J,N) + GGS(J,5,N) + GGS(J,8,N)) 0 5600
      * + Y(LOU1)*(BCOF(6,J,N) + GGS(J,6,N) + GGS(J,9,N)) 0 5601
      * -Y(LOU3)*ACOF(7,J,N) - Y(LOU4)*ACOF(8,J,N) - Y(LOU5)*ACOF(9,J,N) 0 5602
      15 CONTINUE                                  0 5603
      CALL MULTAD (YDT(LO),CW,G(LOU),1,LE,3,1,KM,1) 0 5604
      CALL MULT3 (CGFXY(1,1,N),YDT(LO),CW(1,1),LE,LE,1,KM,1,KM) 0 5605
      CALL MULT3 (CGFXZ(1,1,N),YDT(LO),CW(1,2),LE,LE,1,KM,1,KM) C 5606
      CALL MULT3 (CGFYZ(1,1,N),YDT(LO),CW(1,3),LE,LE,1,KM,1,KM) 0 5607
      CALL MULT3 (YDT(LO),CW,V,1,LE,3,1,KM,1)      0 5608
      G(LOU) = G(LOU) - V(3)                        0 5609
      G(LOU1) = G(LOU1) - V(2)                     0 5610
      G(LOU2) = G(LOU2) - V(1)                     0 5611
C
C      COMPUTE EQUATION II-94,95,96                 0 5612
C
      DO 16 J=1,LE                                  0 5613
      CW(J,1) = -Y(LOU)*ACOF(1,J,N) - Y(LOU1)*ACOF(4,J,N) 0 5614
      * -Y(LOU2)*ACOF(7,J,N)                      0 5615
      CW(J,2) = -Y(LOU)*ACOF(2,J,N) - Y(LOU1)*ACOF(5,J,N) C 5616
      * -Y(LOU2)*ACOF(8,J,N)                      0 5617

```

```

      CW(J,3) = -Y(LDU )*ACOF(3,J,N) - Y(LDU1)*ACOF(6,J,N)      0 5622
      *      -Y(LDU2)*ACOF(9,J,N)                                0 5623
18  CONTINUE                                                    0 5624
      CALL MULTAD (YDT(LD),CW,G(LDU3),1,LE,3,1,KM,1)           0 5625
C                                                                    0 5626
C                                                                    0 5627
C      COMPUTE EQUATION 11-97                                     0 5628
      DO 20 J=1,LE                                             0 5629
      I = LDU5 + J                                             0 5630
      G(I) = G(I) + (Y(LDU )**2)*(BCOF(1,J,N) + GGS(J,1,N))    0 5631
      *      + (Y(LDU1)**2)*(BCOF(2,J,N) + GGS(J,2,N))        0 5632
      *      + (Y(LDU2)**2)*(BCOF(3,J,N) + GGS(J,3,N))        0 5633
      *      - Y(LDU )*Y(LDU1)*(BCOF(4,J,N) + GGS(J,4,N) + GGS(J,7,N)) 0 5634
      *      - Y(LDU )*Y(LDU2)*(BCOF(5,J,N) + GGS(J,5,N) + GGS(J,8,N)) 0 5635
      *      - Y(LDU1)*Y(LDU2)*(BCOF(6,J,N) + GGS(J,6,N) + GGS(J,9,N)) 0 5636
      *      + Y(LDU )*(Y(LDU3)*ACOF(1,J,N) + Y(LDU4)*ACOF(2,J,N) 0 5637
      *      + Y(LDU5)*ACOF(3,J,N))                             0 5638
      *      + Y(LDU1)*(Y(LDU3)*ACOF(4,J,N) + Y(LDU4)*ACOF(5,J,N) 0 5639
      *      + Y(LDU5)*ACOF(6,J,N))                             0 5640
      *      + Y(LDU2)*(Y(LDU3)*ACOF(7,J,N) + Y(LDU4)*ACOF(8,J,N) 0 5641
      *      + Y(LDU5)*ACOF(9,J,N))                             0 5642
20  CONTINUE                                                    0 5643
C                                                                    0 5644
      CALL MULT3 (COFXY(1,1,N),YDT(LD),CW(1,1),LE,LE,1,KM,1,KM) 0 5645
      CALL MULT3 (COFXZ(1,1,N),YDT(LD),CW(1,2),LE,LE,1,KM,1,KM) 0 5646
      CALL MULT3 (COFYZ(1,1,N),YDT(LD),CW(1,3),LE,LE,1,KM,1,KM) 0 5647
      CALL MULT3 (YDT(LD),COFXY(1,1,N),RW(1,1),1,LE,LE,1,KM,3) 0 5648
      CALL MULT3 (YDT(LD),COFXZ(1,1,N),RW(2,1),1,LE,LE,1,KM,3) 0 5649
      CALL MULT3 (YDT(LD),COFYZ(1,1,N),RW(3,1),1,LE,LE,1,KM,3) 0 5650
      DO 30 J=1,LE                                             0 5651
      I = LDU5 + J                                             0 5652
      G(I) = G(I) + Y(LDU )*(CW(J,3) - RW(3,J))                0 5653
      *      + Y(LDU1)*(CW(J,2) - RW(2,J))                    0 5654
      *      + Y(LDU2)*(CW(J,1) - RW(1,J))                    0 5655
30  CONTINUE                                                    0 5656
C                                                                    0 5657
100 NMUN = NMUN(1,N)                                           0 5658
C                                                                    0 5659
C      GET CONTRIBUTIONS DUE TO WHEELS EQUATION 11-109         0 5660
      IF (NMUN.EQ. 0) GO TO 50                                0 5661
      IC = 0                                                    0 5662
      DO 35 I=1,NMUN                                           0 5663
      IP2 = I + 2                                              0 5664
      NW = NMUN(IP2,N)                                         0 5665
C      CHECK TO SEE IF WHEEL NW IS ACTIVE                      0 5666
      IF (IMU(3,NW).EQ. 0) GO TO 37                           0 5667
      IC = IC + 1                                              0 5668
      LMC = LDU5 + LE + IC                                     0 5669
      TDTJ = Y(LMU)*AMU(2,NW)                                  0 5670
      GO TC 38                                                  0 5671
37  TDTJ = AMU(1,NW)*AMU(2,NW)                                  0 5672
38  NPTS = IMU(1,NW)                                           0 5673
      NAX = IMU(2,NW)                                           0 5674
      CALL MULT3 (BS(1,1,NPTS),Y(LDU),V(4),3,LEJ,1,6,1,1)    0 5675
      GO TC (41,42,43), NAX                                    0 5676
41  V(1) = ZRO                                                0 5677
      V(2) = -V(6)*TDTJ                                         0 5678
      V(3) = V(5)*TDTJ                                         0 5679
      GO TC 40                                                  0 5680
42  V(1) = V(6)*TDTJ                                         0 5681

```

V(2) = ZRO	0 5682
V(3) = -V(4)*TDTJ	0 5683
GO TO 40	0 5684
43 V(1) = -V(5)*TDTJ	0 5685
V(2) = V(4)*TDTJ	0 5686
V(3) = ZRO	0 5687
40 CALL MULTAD (V,BS(1,1,NPTS),G(LOU),1,3,LEU,1,3,1)	0 5688
IF (IMO(3,NW) .EQ. 0) GO TO 35	0 5689
G(LMC) = TSHFT(NW)	0 5690
IF (LE .EQ. 0) GO TO 35	0 5691
CALL MULT3 (BS(1,7,NPTS),YDT(LO),V,3,LE,1,6,1,1)	0 5692
CALL SKEW3 (V,WSK,1,3)	0 5693
CALL MULT3 (WSK,V(4),V,3,3,1,3,1,1)	0 5694
G(LMC) = G(LMU) - AMU(2,NW)*V(NAX)	0 5695
35 CONTINUE	0 5696
C	0 5697
50 CONTINUE	0 5698
C	0 5699
CCC SUBROUTINE EQADD ESTABLISHES ADDITIONAL CONTROL BLOCK EQUATIONS	0 5700
CCCC TO SET UP SIMILARITY TRANSFORMATION. USED ONLY FOR LINEARIZATION	0 5701
CCCC AND STABILITY PACKAGE. USER SUPPLIED --	0 5702
CALL EQADD	0 5703
C	0 5704
IF (.NOT.LEQU) RETURN	0 5705
WRITE (NCT,1002)	0 5706
1002 FORMAT (' LOAD VECTOR G WITH GYROSCOPIC EFFECTS ADDED ON')	0 5707
CALL WRITES(G,1,NO,1)	0 5708
RETURN	0 5709
END	0 5710
SUBROUTINE ENGMUM	0 5711
C	0 5712
IMPLICIT REAL*8(A-H,O-Z)	0 5713
C	0 5714
C FOR THEORY OF EQUATIONS PROGRAMMED SEE VOL 1 APPENDIX D	0 5715
COMMON /SHBSRD/	0 5716
* BH(6,18,11),BS(6,18,15),ROL(3,3,6),JUL(3,6)	2 5717
COMMON /INTGR/	0 5718
* AM(171,6),ACCF(9,12,6),BCCF(6,12,6),	5 5719
* CCF11(12,12,6),CCF22(12,12,6),CCF33(12,12,6),AK(12,12,6),	6 5720
* CCF12(12,12,6),CCF13(12,12,6),CCF23(12,12,6),AD(12,12,6),	7 5721
* CCFX1(12,12,6),CCFX2(12,12,6),CCFY2(12,12,6)	8 5722
COMMON /MAXMUM/	0 5723
* NBMX,NHMX,NSPMX,NMWMX,NMWEGD,NMDBUD,KMU,KY,KU	0 5724
COMMON /MUMENG/	0 5725
* P(113),PNUM(36),HTOT(3),TOTL(3),ENGKE(6),ENGPE(6),	11 5726
* TOTKE, TOTPE, TOTENG, AHTOT, ATOTL	0 5727
COMMON /NUMBS/	0 5728
* ZRC,ONE,TWO,THREE	0 5729
COMMON /SPECIF/	0 5730
* DELTAH(6,6),DELTAHD(6,6),AMC(2,5),RH(3,3,30),RS(3,3,30),	16 5731
* CH(3,35),JS(3,30),IMG(3,5),NMOW(6,5),IFISHA(15),	17 5732
* NG,NH,NSPT,NDFMC,NDELTA,ITCPOL(2,6),IRGFLX(6),IHDATA(7,6),	18 5733
* LCC(14),LENU(14),NU,NBETA,NLAM,NEQ	19 5734
COMMON /VECTOR/	0 5735
DEBUG # 58	0 5712

	* Y(250),YDT(250)	20 5736
C	DIMENSION WV(12)	0 5737
	DIMENSION VW(3)	89 5738
		0 5739
C	KM = NMC80D	0 5740
	DO 5 I=1,3	0 5741
	HTOT(I) = ZRO	0 5742
	5 TOTL(I) = ZRO	0 5743
	DO 10 N=1,NB	0 5744
	LA = 6*N - 5	0 5745
	LL = LA + 3	0 5746
	LQA = LQCU(N)	0 5747
	LQL = LQCU(N) + 3	0 5748
	CALL MULT3 (ROL(1,1,N),P(LQA),PMCM(LA),3,3,1,3,1,1)	0 5749
	CALL MULT3 (ROL(1,1,N),P(LQL),PMCM(LL),3,3,1,3,1,1)	0 5750
	PMCM(LA) = PMCM(LA) + DOL(2,N)*PMCM(LL+2) - DOL(3,N)*PMCM(LL+1)	0 5751
	PMCM(LA+1) = PMCM(LA+1) + DOL(3,N)*PMCM(LL) - DOL(1,N)*PMCM(LL+2)	0 5752
	PMCM(LA+2) = PMCM(LA+2) + DOL(1,N)*PMCM(LL+1) - DOL(2,N)*PMCM(LL)	0 5753
CCC	STATEMENTS THRU 40 TO ACCOUNT FOR ANGULAR MOMENTUM DUE TO	0 5754
CCC	CONSTANT SPEED MOMENTUM WHEELS.	0 5755
	NM = NMOW(1,N)	0 5756
	IF (NM .EQ. 0) GO TO 40	0 5757
	DO 30 I=1,NM	0 5758
	NW = NMOW(2+I,N)	0 5759
	IF (IMG(3,NW) .NE. 0) GO TO 30	0 5760
	NA = IMG(2,NW)	0 5761
	NS = IMG(1,NW)	0 5762
	PH = AMG(1,NW)*AMG(2,NW)	0 5763
	DO 35 J=1,3	0 5764
35	VW(J) = PH*BS(NA,J,NS)	0 5765
	CALL MULTAD (ROL(1,1,N),VW,PMCM(LA),3,3,1,3,1,1)	0 5766
30	CONTINUE	0 5767
40	CONTINUE	0 5768
	DO 15 I=1,3	0 5769
	I1 = LA - 1 + I	0 5770
	I2 = LL - 1 + I	0 5771
	HTOT(I) = HTOT(I) + PMCM(I1)	0 5772
15	TOTL(I) = TOTL(I) + PMCM(I2)	0 5773
10	CONTINUE	0 5774
C		0 5775
	TOTKE = ZRO	0 5776
	DO 20 N=1,NB	0 5777
	LQU = LQCU(N)	0 5778
	LO = LQCU(N+NB)	0 5779
	ENGPE(N) = ZRO	0 5780
	LE = LQCU(N+NB)	0 5781
	LEU = LQCU(N)	0 5782
	IF (LE .EQ. 0) GO TO 22	0 5783
	CALL MULT3 (AK(1,1,N),Y(LO),WV,LE,LE,1,KM,1,1)	0 5784
	CALL MULT3 (Y(LO),WV,ENGPE(N),1,LE,1,1,1,1)	0 5785
	ENGPE(N) = ENGPE(N)/TWO	0 5786
22	CALL MULT3 (Y(LQU),P(LQU),ENGKE(N),1,LEU,1,1,1,1)	0 5787
	ENGKE(N) = ENGKE(N)/TWO	0 5788
	TOTKE = TOTKE + ENGKE(N)	0 5789
20	TOTPE = TOTPE + ENGPE(N)	0 5790
	TCTENG = TOTKE + TOTPE	0 5791
	ATOTL = DSQRT(TOTL(1)**2 + TOTL(2)**2 + TOTL(3)**2)	0 5792
	HTOT = DSQRT(HTOT(1)**2 + HTOT(2)**2 + HTOT(3)**2)	0 5793
C		0 5794
	RETURN	0 5795
	END	0 5796
		0 5797

	SUBROUTINE RKADAM(NEQ)		0 5798
C		DEBUG # 59	0 5799
	IMPLICIT REAL*8(A-H,O-Z)		0 5800
C			0 5801
	COMMON /JILFLG/		0 5802
*	JIL		0 5803
	COMMON /PRWORK/		0 5804
*	PR(250,5)	14	5805
	COMMON /QPKKTA/		0 5806
*	QPKK(250),PRK(4), NT	15	5807
	COMMON /TIMESS/		0 5808
*	STARTT,DELTAT,T,ENDT,TMST		0 5809
	COMMON /VECTOR/		0 5810
*	Y(250),YDT(250)	20	5811
	COMMON /VINDEP/		0 5812
*	INDEP(250)	21	5813
C			0 5814
	DATA EPS1, EPS2 / 1.0-8, 1.0-2 /		0 5815
	DATA NDT,MAXIT / 0, 10/		0 5816
C			0 5817
C	***** ITYPE .EQ. 1 RUNGE-KUTTA INTEGRATION		0 5818
C	***** ITYPE .EQ. 2 ADAMS PREDICTOR/CORRECTOR INTEGRATION		0 5819
C			0 5820
C	SUBROUTINE TO INTEGRATE DIFFERENTIAL EQUATIONS (FIRST ORDER)		0 5821
C	IN THE TIME DOMAIN. USES RUNGE-KUTTA-GILL TO START THE ADAMS PREDICT		0 5822
C	CORRECTOR. MAY USE RUNGE-KUTTA ONLY, ON OPTION.		0 5823
C	CODED BY CARL BODLEY 1971		0 5824
C	MODIFICATION TO CORRECTOR LOOP TO ACCOUNT FOR IMPULSIVE CHANGE TO		0 5825
C	STATE VECTOR (0)WHICH OCCURS IN SUB. YDOF. MADE BY CARL BODLEY		0 5826
C	APR, 1974		0 5827
C			0 5828
	DATA ITYPE / 1 /		0 5829
C			0 5830
	GO TO (10,20) , ITYPE		0 5831
C			0 5832
20	IF (NT .GT. 0) GO TO 201		0 5833
	DNM = DELTAT/24.0 0		0 5834
	TR1 = DNM*55.0 0		0 5835
	TR2 = -DNM*59.0 0		0 5836
	TR3 = DNM*37.0 0		0 5837
	TR4 = -DNM* 9.0 0		0 5838
	TR5 = DNM* 9.0 0		0 5839
	TR6 = DNM*19.0 0		0 5840
	TR7 = -DNM* 5.0 0		0 5841
	TR8 = DNM* 1.0 0		0 5842
C			0 5843
201	IF (NT .GT. 3) GO TO 200		0 5844
	NL = NT + 1		0 5845
	GO 2CS I=1,NEQ		0 5846
	PR(I,NL) = YDT(I)		0 5847
205	PR(I, 5) = Y (I)		0 5848
	IF (NT .EQ. 3) GO TO 200		0 5849

C		0 5850
	10 CO 120 J = 1,4	0 5851
	JIL = J	0 5852
	CO 110 I=1,NEQ	0 5853
	Z = YDT(1)*DELTAT	0 5854
	GO TC (103,101,101,105), JIL	0 5855
	101 R = FRK(JIL)*(Z - QRK(1))	0 5856
	GO TC 107	0 5857
	103 R = FRK(JIL)*Z - QRK(1)	0 5858
	GO TC 107	0 5859
	105 R = (Z - 2.0 0*QRK(1))/6.0 0	0 5860
	107 Y(1) = Y(1) + R	0 5861
	QRK(1) = QRK(1) + 3.0 0*R - PRK(JIL)*Z	0 5862
	110 CONTINUE	0 5863
	IF (JIL .EQ. 1 .OR. JIL .EQ. 3) T = T + DELTAT/2.0 0	0 5864
	120 CALL YDOT	0 5865
	GO TC 300	0 5866
C		0 5867
	200 CO 204 I=1,NEQ	0 5868
	204 Y(1) = PR(1,5)+TR1*PR(1,4)+TR2*PR(1,3)+TR3*PR(1,2)+TR4*PR(1,1)	0 5869
	T = T + DELTAT	0 5870
	ITER = 0	0 5871
	207 CALL YDOT	0 5872
C		0 5873
	G = C.D C	0 5874
	CO 203 I=1,NEQ	0 5875
	IF (INDEP(1) .EQ. 0) GO TO 203	0 5876
	YC = PR(1,5)+TR5*YDT(1) +TR6*PR(1,4)+TR7*PR(1,3)+TR8*PR(1,2)	0 5877
	DN = DABS(Y(1))	0 5878
	DN1 = DABS(YC)	0 5879
	IF (DN1 .GT. DN) DN = DN1	0 5880
	IF (DN .LT. EPS1) GO TO 203	0 5881
	G1 = DABS(YC - Y(1))/DN	0 5882
	IF (G1 .GT. G) G = G1	0 5883
	Y(1) = YC	0 5884
	203 CONTINUE	0 5885
	ITER = ITER + 1	0 5886
	IF (G .LE. EPS2) GO TO 30	0 5887
	IF (ITER .EQ. MAXIT) GO TO 999	0 5888
	GO TC 207	0 5889
C		0 5890
	30 CO 210 I=1,NEQ	0 5891
	FR(1,1) = PR(1,2)	0 5892
	PR(1,2) = PR(1,3)	0 5893
	PR(1,3) = PR(1,4)	0 5894
	PR(1,4) = YDT(1)	0 5895
	210 PR(1,5) = Y(1)	0 5896
C		0 5897
	300 NT = NT + 1	0 5898
	ANT = NT	0 5899
	TMST = ANT*DELTAT	0 5900
	T = STARTT + TMST	0 5901
C		0 5902
	RETURN	0 5903
	999 WRITE (NGT,1001) MAXIT	0 5904
	1001 FORMAT (1H1,31HCONVERTOR FAILS TO CONVERGE IN 13,	0 5905
	* 28HITERATIONS, PROGRAM STOPPED.)	0 5906
	STOP	0 5907
	END	0 5908

```

SUBROUTINE PRINTOU                                0 5909
C                                                    0 5910
IMPLICIT REAL*8(A-H,C-Z)                          0 5911
REAL*4 CTNEW,TNEW,CTOLD,TOLD,CPSTEP,CPSEC          0 5912
C                                                    0 5913
COMMON /AMUBW /                                    0 5914
*   AMU(22,22, 6),BW(30,113)                      1 5915
COMMON /BHBSR /                                    0 5916
*   BH(6,16,11),BS(6,18,15),ROL(3,3, 6),JOL(3, 6) 2 5917
COMMON /MAXMUM /                                    0 5918
*   NMAX,NHMAX,NPMAX,NMWMAX,NMWBOO,NMDBOO,K4J,KY,KU 0 5919
COMMON /LAMBDA /                                    0 5920
*   ALAM(J6)                                         10 5921
COMMON /MISCNU /                                    0 5922
*   NUPFNT,NUPLOT                                    0 5923
COMMON /MLMENS /                                    0 5924
*   P(113),PMOM(36),HTOT(3),TCTL(3),ENGKE( 6),ENGPE( 6), 11 5925
*   TOTKE, TOTPE, TOTENG, AHTGT,ATOTL               0 5926
COMMON /SPECIF /                                    0 5927
*   BETAH(6, 6),BETAHD(6, 6),AMO(2, 5),RH(3,3,30),RS(3,3,30), 16 5928
*   CH(3,35),DS(3,30),IMC(3, 5),NMOH(6, 6),IFTS,MW(15), 17 5929
*   NB,NH,NSPT,NCFMU,NDELTA,ITCPOL(2, 6),IRGFLX( 6),IHDATA(7, 6), 18 5930
*   LCCU(14),LENU(14),NU,NBETA,NLAM,NEO            19 5931
COMMON /TIMESS /                                    0 5932
*   STARTT,DELTA,T,ENDT,TMST                       0 5933
COMMON /VECTOR /                                    0 5934
*   Y(250),YDT(250)                                20 5935
C                                                    0 5936
DIMENSION V1(3), V2(3)                             0 5937
DIMENSION T1(3,3), T2(3,3), TRK(3,3)              0 5938
DIMENSION VCM(3, 6),TIN(3,3, 6),SYSCM(3),SYSIN(3,3),XMAS( 6) 93 5939
DATA NJT,LIST / 6, 0 /                             0 5940
C                                                    0 5941
1000 FORMAT (//10X,24HAT SIMULATION TIME, T = ,1P10.4,32(2H* )) 0 5942
1001 FORMAT ( 3X,21HTHE STATE VECTOR Y = )          0 5943
1002 FORMAT ( 3X,39HTHE STATE VECTOR TIME DERIVATIVE YDT = ) 0 5944
1003 FORMAT ( 3X,50HTHE BETAS (EULER ANGLES, POSITION COORDINATES) ARE) 0 5945
1004 FORMAT ( 3X,29HTHE BETA TIME DERIVATIVES ARE) 0 5946
1005 FORMAT ( 3X,41HTHE DELTAS (CONTROL SYSTEM VARIABLES) ARE) 0 5947
1006 FORMAT ( 3X,30HTHE DELTA TIME DERIVATIVES ARE) 0 5948
1007 FORMAT ( 3X, 9HFOR BODY ,12,3X,18HTHE VELOCITIES ARE) 0 5949
1017 FORMAT ( 3X, 9HFOR BODY ,12,3X,29HTHE CORRESPONDING MOMENTA ARE) 0 5950
1027 FORMAT ( 3X, 9HFOR BODY ,12,3X,25HITS CONTRIBUTION TO TOTAL, 0 5951
*   21H ANGULAR AND LINEAR MOMENTUM IS)              0 5952
1008 FORMAT ( 3X,48HITS CONTRIBUTION TO TOTAL KINETIC AND POTENTIAL , 0 5953
*   12HENERGIES IS ,3X,1P2015.8)                    0 5954
1009 FORMAT ( 3X, 9HFOR BODY ,12,3X,27HTHE ELASTIC DEFLECTIONS ARE) 0 5955
1010 FORMAT ( 3X,50HTHE INTERCONNECTION CONSTRAINT FORCES(LAMBDAS) ARE) 0 5956
1011 FORMAT ( 3X,30HTHE TOTAL ANGULAR MOMENTUM VECTOR IS) 0 5957
1012 FORMAT ( 3X,35HTHE TOTAL LINEAR MOMENTUM VECTOR IS) 0 5958
1013 FORMAT (/3X,29HTHE TOTAL ANGULAR MOMENTUM = ,1P15.8, 0 5959
*   / 3X,29HTHE TOTAL LINEAR MOMENTUM = ,1P15.8,    0 5960
*   / 3X,29HTHE TOTAL KINETIC ENERGY = ,1P15.8,   0 5961
*   / 3X,29HTHE TOTAL POTENTIAL ENERGY = ,1P15.8, 0 5962
*   / 3X,29HTHE TOTAL ENERGY (T + V) = ,1P15.8)   0 5963
1014 FORMAT (//35X,33HCPU TIME/STEP CPU TIME/REAL TIME, /38X,1P10.4, 0 5964
*   9X,1P10.4)                                       0 5965
C                                                    0 5966
IF (LIST.EQ. 1) GO TO 5                             0 5967
CTNEW = 0.                                           0 5968

```

	TNEW = STARTT	0 5969
5	CALL PAGEHD	0 5970
	WRITE (NCT,1000) T	0 5971
	WRITE (NCT,1001)	0 5972
	CALL WRITES (Y,1,NEQ,1)	0 5973
C		0 5974
	WRITE (NCT,1000) T	0 5975
	WRITE (NCT,1002)	0 5976
	CALL WRITES (YDT,1,NEQ,1)	0 5977
C		0 5978
	WRITE (NCT,1000) T	0 5979
	WRITE (NCT,1003)	0 5980
	CALL WRITES (BETAH,6,NH,6)	0 5981
C		0 5982
	WRITE (NCT,1000) T	0 5983
	WRITE (NCT,1004)	0 5984
	CALL WRITES (BETAHD,6,NH,6)	0 5985
C		0 5986
	IF (NDELTA .EQ. 0) GO TO 10	0 5987
	WRITE (NCT,1000) T	0 5988
	WRITE (NCT,1005)	0 5989
	LO = LUCU(2*NB + 2)	0 5990
	CALL WRITES (Y(LO),1,NDELTA,1)	0 5991
	WRITE (NCT,1000) T	0 5992
	WRITE (NCT,1006)	0 5993
	CALL WRITES (YDT(LO),1,NDELTA,1)	0 5994
C		0 5995
10	DO 20 N=1,NB	0 5996
	WRITE (NCT,1000) T	0 5997
	WRITE (NCT,1007) N	0 5998
	LO = LUCU(N)	0 5999
	LE = LENG(N)	0 6000
	CALL WRITES (Y(LO),1,LE,1)	0 6001
	WRITE (NCT,1017) N	0 6002
	CALL WRITES (P(LO),1,LE,1)	0 6003
	LQPM = C*(N-1) + 1	0 6004
	WRITE (NCT,1027) N	0 6005
	CALL WRITES (PMQM(LQPM),1,6,1)	0 6006
	WRITE (NCT,1008) ENGKL(N),ENGPE(N)	0 6007
	LE = LENG(N+NB)	0 6008
	IF (LE .EQ. 0) GO TO 31	0 6009
	LO = LUCU(N+NB)	0 6010
	WRITE (NCT,1009) N	0 6011
	CALL WRITES (Y(LO),1,LE,1)	0 6012
C		0 6013
C	BODY N TO BODY 1 TRANSFORMATION MATRIX	0 6014
31	DO 21 I=1,3	0 6015
	DO 21 J=1,3	0 6016
	TRN(I,J) = 0.000	0 6017
	DO 21 K=1,3	0 6018
	TRN(I,J) = TRN(I,J) + ROL(K,I,1)*ROL(K,J,N)	0 6019
21	CONTINUE	0 6020
	WRITE (NCT,1020) N	0 6021
1020	FORMAT (' 3X, 9HFOR BODY ,I2,3X,46HTRANSFORMATION MATRIX TO BODY 1	0 6022
	*COORDINATES IS)	0 6023
	CALL WRITES (TRN,3,3,3)	0 6024
C		0 6025
C	LOCATION BODY N CM WRT BODY 1 REFERENCE	0 6026
	LE = LENG(N)	0 6027
	CALL INVINP(AMU(1,1,N),AMU(1,1,N),LE,KMU)	0 6028

```

      XMAS(N) = AMU(4,4,1)                                0 6029
      V1(1) = AMU(3,3,N)/XMAS(N)                          0 6030
      V1(2) = AMU(1,6,N)/XMAS(N)                          0 6031
      V1(3) = AMU(2,4,N)/XMAS(N)                          0 6032
      DO 22 I=1,3                                          0 6033
      VCM(1,N) = 0.000                                     0 6034
      DO 22 K=1,3                                          0 6035
      VCM(1,N) = VCM(1,N) + TRN(1,K)*V1(K)                0 6036
      *              + KGL(K,1,1)*(DGL(K,N)-DGL(K,1))      0 6037
22 CONTINUE                                              0 6038
      WRITE (NCT,1015) N                                  0 6039
1015 FORMAT ( 3X, 9HFOR BODY ,12,3X,57HCM LOCATION WRT BODY 1 REFERENCE 0 6040
      * POINT AND COORDINATES IS)                          0 6041
      CALL WRITES (VCM(1,N),1,3,1)                        0 6042
C                                                    0 6043
C      INERTIA TENSOR OF BODY N ABOUT ITS CM WRT BODY 1 REFERENCE FRAME 0 6044
      DO 23 I=1,3                                          0 6045
      DO 23 J=1,3                                          0 6046
      T1(I,J) = AMU(I,J,N)                                0 6047
      DO 23 K=1,3                                          0 6048
      T1(I,J) = T1(I,J) - AMU(I,K+3,N)*AMU(K+3,J,N)/XMAS(N) 0 6049
23 CONTINUE                                              0 6050
C                                                    0 6051
C      INERTIA TENSOR OF BODY N ABOUT ITS CM WRT BODY 1 REFERENCE 0 6052
      DO 24 I=1,3                                          0 6053
      DO 24 J=1,3                                          0 6054
      TIN(I,J,N) = 0.000                                   0 6055
      DO 24 K=1,3                                          0 6056
      DO 24 L=1,3                                          0 6057
      TIN(I,J,N) = TIN(I,J,N) + TRN(I,K)*T1(K,L)*TRN(J,L) 0 6058
24 CONTINUE                                              0 6059
      WRITE (NCT,1016) N                                  0 6060
1016 FORMAT ( 3X, 9HFOR BODY ,12,3X,71HINERTIA TENSOR RELATIVE TO ITS C 0 6061
      *ENTER OF MASS WRT BODY 1 COORDINATES IS)            0 6062
      CALL WRITES (TIN(1,1,N),3,3,3)                      0 6063
      CALL INVNP(AMU(1,1,N),AMU(1,1,N),LE,KMU)            0 6064
C                                                    0 6065
20 CONTINUE                                              0 6066
C                                                    0 6067
      IF (NLAM .EQ. 0) GO TO 50                            0 6068
      WRITE (NCT,1000) T                                  0 6069
      WRITE (NCT,1010)                                     0 6070
      CALL WRITES (ALAM,1,NLAM,1)                           0 6071
C                                                    0 6072
50 WRITE (NCT,1000) T                                    0 6073
C                                                    0 6074
C      COMPOSITE SYSTEM CENTER OF MASS AND INERTIA TENSOR RELATIVE TO IT 0 6075
C      IN BODY 1 COORDINATES                               0 6076
      XMASS = 0.00                                         0 6077
      DO 25 I=1,3                                          0 6078
      SYSCM(I) = 0.000                                     0 6079
      DO 25 J=1,3                                          0 6080
      SYSIN(I,J) = 0.000                                   0 6081
25 CONTINUE                                              0 6082
C                                                    0 6083
      DO 26 N=1,NB                                         0 6084
      XMASS = XMASS + XMAS(N)                              0 6085
      DO 26 I=1,3                                          0 6086
      SYSCM(I) = SYSCM(I) + VCM(1,N)*XMAS(N)              0 6087
      DO 26 J=1,3                                          0 6088

```

	SYN(1,J) = SYN(1,J) + T(1,J,N)	0 6089
26	CONTINUE	0 6090
	DO 27 I=1,3	C 6091
27	SYSCM(1) = SYSCM(1)/XMASS	0 6092
C		0 6093
	DO 28 N=1,NB	0 6094
	DO 29 I=1,3	0 6095
29	V1(I) = VCM(1,N) - SYSCM(1)	0 6096
	CALL SKEWVJ(V1,T1,3,3)	0 6097
	DO 30 I=1,3	0 6098
	DO 30 J=1,3	0 6099
	DO 30 K=1,3	0 6100
	SYN(1,J) = SYN(1,J) + XMAS(N)*T1(I,K)*T1(J,K)	C 6101
30	CONTINUE	0 6102
28	CONTINUE	0 6103
C		0 6104
	WRITE (NCT,1018)	C 6105
1018	FORMAT (3X,79HTHE SYSTEM CENTER OF MASS RELATIVE TO BODY 1 REFERE	0 6106
	*NCE POINT AND COORDINATES IS)	0 6107
	CALL WRITES(SYSCM,1,3,1)	0 6108
	WRITE (NCT,1019)	0 6109
1019	FORMAT (3X,84HTHE SYSTEM INERTIA TENSOR RELATIVE TO SYSTEM CENTER	0 6110
	* OF MASS IN BODY 1 COORDINATES IS)	0 6111
	CALL WRITES(SYN,3,3,3)	0 6112
C		0 6113
	WRITE (NCT,1011)	0 6114
	CALL WRITES (HTOT,1,3,1)	0 6115
	WRITE (NCT,1012)	0 6116
	CALL WRITES (TOTL,1,3,1)	0 6117
C		0 6118
	WRITE (NCT,1013) AHCT,ATOTL,TCTKE,TOTPE,TJTEG	C 6119
C		0 6120
	IF (IIST.EQ. 1) GO TO 100	0 6121
	IIST = 1	0 6122
	RETURN	0 6123
100	TOLD = TNEW	0 6124
	CTOLD = CTNEW	0 6125
	TNEW = T	0 6126
	CALL SECCND (CTNEW)	0 6127
	CPSTEP = CTNEW - CTOLD	0 6128
	CPSEC = CPSTEP/(TNEW-TOLD)	0 6129
	CPSTEP = CPSTEP/FLUAT(NOPRNT)	0 6130
	WRITE (NCT,1014) CPSTEP,CPSEC	0 6131
C		0 6132
	RETURN	0 6133
	END	0 6134
	SUBROUTINE PLOTWR	0 6135
C		0 6136
	IMPLICIT REAL*8(A-H,O-Z)	0 6137
C		0 6138
C	SUBROUTINE TO WRITE PLOT OUTPUT TAPE	C 6139
C		0 6140
	COMMON /LAMBDA/	0 6141
	* ALAM(36)	10 6142

DATA KRPLUT,KCPLUT /1000,16/	91 6197
C	0 6198
READ (NIT,1005) (ICTITL(I),I=1,10)	0 6199
1005 FORMAT(10A8)	0 6200
CALL PAGEHD	0 6201
WRITE (NCT,1001) (ICTITL(I),I=1,10)	0 6202
1001 FORMAT (///,30X,31HSUMMARY OF PLOTTING INFORMATION//,10X,10A8)	0 6203
C	0 6204
READ (NIT,1003) NSET	0 6205
1003 FORMAT(16I5)	0 6206
IF (NSET .EQ. 0) RETURN	0 6207
WRITE (NCT,1011) NSET,NRPLUT,NCPLUT,KRPLUT,KCPLUT	0 6208
1011 FORMAT(//,10X,10HNSET =15,/,	0 6209
* 10X,10HNRPLUT =15, 10X,10HNCPLUT =15,/,	0 6210
* 10X,10HKRPLUT =15, 10X,10HKCPLUT =15)	0 6211
C	0 6212
IF (NRPLUT .LE. KRPLUT) GO TO 1500	0 6213
NRPLUT = KRPLUT	0 6214
WRITE (NCT,1009)	0 6215
1009 FORMAT (//,10X,46HNRPLUT EXCEEDED KRPLUT AND WAS RESET TO KRPLUT)	0 6216
1500 CONTINUE	0 6217
C	0 6218
DO 1000 ISET=1,NSET	0 6219
REWIND NTAPE3	0 6220
READ (NIT,1003) JPL	0 6221
IF (JPL .GT. KCPLUT) GO TO 998	0 6222
READ (NIT,1003) (JVPL(J),J=1,JPL)	0 6223
WRITE (NCT,1012) ISET,(JVPL(J),J=1,JPL)	0 6224
1012 FORMAT(//,10X,7HISET = 15,/,10X,7HJVPL =1615)	0 6225
C	0 6226
DO 2000 II=1,NRPLUT	0 6227
READ (NTAPE3) (DUM(I),I=1,NCPLUT)	0 6228
DO 2001 J=1,JPL	0 6229
JC = JVPL(J)	0 6230
2001 ZP(II,J) = DUM(JC)	0 6231
2000 CONTINUE	0 6232
C	0 6233
20 READ (NIT,1003) NCI,(NCD(I),I=1,3),NGRID	0 6234
IF (NCI .EQ. 0) GO TO 1000	0 6235
IF(NCI .GT. 1) NGRID = 1	0 6236
IF(NGRID .EQ. 0) NGRID = 1	0 6237
READ (NIT,1004) TITL1,TITL2,(PTITLE(I),I=1,6)	0 6238
1004 FORMAT(2(A8,2X),6A8)	0 6239
WRITE(NCT,1006) NCI,(NCD(I),I=1,3),NGRID,	0 6240
* TITL1,TITL2,(PTITLE(I),I=1,6)	0 6241
1006 FORMAT(//,15X,7HNCD =,15,5X,7HNCD =,315,5X,7HNGRID =,15,/,	0 6242
* 15X,A8,5X,A8,5X,6A8)	0 6243
CALL PLTCAR(ZP,NCI,NCD,NRPLUT,NGRID,	0 6244
* TITL1,TITL2,PTITLE,ICTITL,KRPLUT)	0 6245
GO TO 20	0 6246
C	0 6247
1000 CONTINUE	0 6248
C	0 6249
RETURN	0 6250
C	0 6251
998 WRITE (NCT,1020)	0 6252
1020 FORMAT(//,10X,34HERROR IN PLOT INPUT DATA, STOP RUN)	0 6253
STOP	0 6254
C	0 6255
END	0 6256

```

SUBROUTINE PLTCAR (DATA,NI,ND,NR,NG,
C
C          *          TITL1,TITL2,TITLP,TITLM,KR)          DEBUG # 63
C
C DATA          DATA ARRAY
C NI             LOCATION OF INDEPENDENT VARIABLE (X)
C ND             LOCATION OF DEPENDENT VARIABLES (Y(I),I=1,3)
C NR             NO OF VALUES TO PLOT
C NG             NO OF GRIDS
C TITL1          TITLE FOR INDEPENDENT VARIABLE AXIS
C TITL2          TITLE FOR DEPENDENT VARIABLE AXIS
C TITLP          PLOT TITLE - UNIQUE TO FRAME
C TITLM          PLOT TITLE - ALL FRAMES
C KR             ROW DIMENSION OF DATA ARRAY IN CALLING PROGRAM
C
REAL*4 MINI,MAXI,MIND,MAXD
REAL*8 TITLP,TITLM,TITL1,TITL2,TITL(5),IRUNNO,IDATE
REAL*8 UNAME,TITLE1,TITLE2
C
COMMON /LZMODE/ AMODE(200)
COMMON /LSTART/ IRUNNO,IDATE,NPAGE
COMMON /L3TRT1/ UNAME(3),TITLE1(12),TITLE2(12)
C
DIMENSION DATA(KR,1),TITLP(1),TITLM(1),ND(1)
DIMENSION VI(1000),VD(1000)
C
DATA NKP /12/
C
FORM TITL
TITL(1) = IRUNNO
TITL(2) = IDATE
TITL(3) = UNAME(1)
TITL(4) = UNAME(2)
TITL(5) = UNAME(3)
C
NO OF PLOTS
NPLOTS = 0
DO 3 I=1,3
IF(ND(I) .NE. 0) NPLOTS = NPLOTS+1
3 CONTINUE
C
FIND MAX/MIN OF DEPENDENT VARIABLES
J = ND(1)
MAXD = DATA(1,J)
MIND = DATA(1,J)
DO 10 L=1,NPLOTS
J = ND(L)
DO 10 I=1,NR
IF(DATA(I,J) .GT. MAXD) MAXD = DATA(I,J)
IF(DATA(I,J) .LT. MIND) MIND = DATA(I,J)
10 CONTINUE
IF(MAXD .EQ. MIND) MAXD = MIND + 10.0
C
GRID LUCK *****
C
NLFT = 1
NDIV = NR/NG
C
DO 45 I1=1,NG
IF(I1 .GT. 1) NLFT=NRGT

```


	NRGT = II*NDIV	0 6317
	IF(II.EQ. NG) NRGT=NR	0 6318
	NP = NRGT - NLFT + 1	0 6319
C		0 6320
C	FIND MAX/MIN OF INDEPENDENT VARIABLE	0 6321
	MAXI = DATA(NLFT,NI)	0 6322
	MINI = DATA(NLFT,NI)	0 6323
	DO 11 I=NLFT,NRGT	0 6324
	IF(DATA(I,NI).GT. MAXI) MAXI = DATA(I,NI)	0 6325
	IF(DATA(I,NI).LT. MINI) MINI = DATA(I,NI)	0 6326
11	CONTINUE	0 6327
	IF(MAXI.EQ. MINI) MAXI = MINI + 10.0	0 6328
C		0 6329
C	INITIALIZE MODE SETS	0 6330
	CALL RSETMG(AMODE)	0 6331
C		0 6332
C	SET OBJECT SPACE	0 6333
	CALL SETSMG(AMODE,19,1.0)	0 6334
	CALL SETSMG(AMODE,20,1.0)	0 6335
	CALL OBJCTG(AMODE,0.0,0.0,1.0,1.0)	0 6336
C		0 6337
C	EQX FRAME	0 6338
	CALL SETSMG(AMODE,14,3.00)	0 6339
	CALL SETSMG(AMODE,30,4.00)	0 6340
	CALL LINESG(AMODE,0.0,0.00,0.00)	0 6341
	CALL LINESG(AMODE,1.0,0.00,0.99)	0 6342
	CALL LINESG(AMODE,1.1,0.00,0.99)	0 6343
	CALL LINESG(AMODE,1.1,0.00,0.00)	0 6344
	CALL LINESG(AMODE,1.0,0.00,0.00)	0 6345
C		0 6346
C	TITLE FRAME	0 6347
	CALL SETSMG(AMODE,45,1.25)	0 6348
	CALL LINESG(AMODE,0.0,0.050,0.020)	0 6349
	CALL TEXTG (AMODE,8,TITL(1))	0 6350
	CALL TEXTG (AMODE,8,TITL(2))	0 6351
	CALL LINESG(AMODE,0.0,0.300,0.020)	0 6352
	CALL TEXTG (AMODE,8,TITL(3))	0 6353
	CALL TEXTG (AMODE,8,TITL(4))	0 6354
	CALL TEXTG (AMODE,8,TITL(5))	0 6355
	CALL LEGNDG(AMODE,0.050,0.00,80,TITLM)	0 6356
	CALL LEGNDG(AMODE,0.300,0.10,48,TITLP)	0 6357
C		0 6358
C	SET OBJECT/SUBJECT SPACE	0 6359
	CALL OBJCTG(AMODE,0.01,0.05,1.0,1.0)	0 6360
	CALL SUBJEG(AMODE,MINI,MIND,MAXI,MAXD)	0 6361
C		0 6362
C	CARTESIAN GRID AND LABELS	0 6363
	CALL SETSMG(AMODE,14,0.00)	0 6364
	CALL SETSMG(AMODE,45,1.00)	0 6365
	CALL SETUPG(AMODE,1,01.00,11TH,IDTH,DLI,DLD,FMTI,FMTD)	0 6366
	DO 12 I=106,109	0 6367
12	AMODE(I) = 0.0	0 6368
	CALL GRIDG (AMODE, D1,DD,11TH,IDTH)	0 6369
	CALL LABELG(AMODE,0,DLI,0, FMTI)	0 6370
	CALL LABELG(AMODE,1,DLD,0, FMTD)	0 6371
	CALL TITLEG(AMODE,8,TITL(1,8,TITLD,0,DUMMY)	0 6372
C		0 6373
C	FLCT DATA	0 6374
	CALL SETSMG(AMODE,30,2.00)	0 6375
	CALL SETSMG(AMODE,45,1.25)	0 6376

DO 40 J=1,NPLOTS	0 6377
I = ND(J)	0 6378
IF(NPLOTS .EQ. 1) GO TO 41	0 6379
AI = DATA(NLEFT,NI)	C 6380
AD = DATA(NLEFT, I)	0 6381
CALL NUMERG(AMODE,AI,AD,1,J)	0 6382
CALL NUMERG(AMODE,AI,AD,1,J)	0 6383
41 NL = NLEFT - 1	0 6384
NK = NP/NKP	0 6385
KK = 0	0 6386
DO 20 K=1,NP	0 6387
VI(K) = DATA(NL+K,NI)	0 6388
VD(K) = DATA(NL+K, I)	0 6389
IF(NPLOTS .EQ. 1) GO TO 20	0 6390
KK = KK + 1	0 6391
IF(KK .NE. NK) GO TO 20	0 6392
KK = 0	0 6393
CALL NUMERG(AMODE,VI(K),VD(K),1,J)	0 6394
CALL NUMERG(AMODE,VI(K),VD(K),1,J)	0 6395
20 CONTINUE	0 6396
CALL LINESG(AMODE,NP,VI,VD)	C 6397
CALL LINESG(AMODE,NP,VI,VD)	0 6398
IF(NPLOTS .EQ. 1) GO TO 40	0 6399
AI = DATA(NRGT,NI)	C 6400
AD = DATA(NRGT, I)	0 6401
CALL NUMERG(AMODE,AI,AD,1,J)	0 6402
CALL NUMERG(AMODE,AI,AD,1,J)	0 6403
40 CONTINUE	0 6404
	0 6405
C ADVANCE FRAME	0 6406
C CALL PAGEG(AMODE,0,1,1)	C 6407
45 CONTINUE	0 6408
	0 6409
C RETURN	0 6410
C END	0 6411
SUERCTINE DYN540	0 6412
C	0 6413
IMPLICIT REAL*8(A-H,O-Z)	0 6414
REAL*4 SAVED, SAVLP, SAVED, SAVEA	0 6415
REAL*4 FMIN,FMAX,DLMIN,DBMAX,TITLE	0 6416
REAL*4 AMIN,AMAX	C 6417
C	0 6418
C ---LATEST MOD ON 09/04/75	C 6419
C	0 6420
C THIS OVERLAY PERFORMS THE LINEARIZED SYSTEM ANALYSES.	C 6421
C	0 6422
C	0 6423
C	0 6424
C -----DATA STREAM CONTROL-----	C 6425
C FOR THIS OVERLAY	0 6426
C	0 6427
C-----LNAM	0 6428
C	0 6429
C IF (LNAM .EQ. 4H) RETURN	C 6430

C	IF (LNAM .EQ. 4HTIME) GO TO 400	0 6431
C		0 6432
C	-----CALL READIM (LRY,10,NCYC,10,KR)	0 6433
C		0 6434
C	NOTE-----LRY(1,J) = ITYPE	0 6435
C	LRY(2,J) = ITFIN	0 6436
C	LRY(3,J) = JTFOUT	0 6437
C	LRY(4,J) = KPLOT	0 6438
C	LRY(5,J) = IAFLG	0 6439
C	LRY(6,J) = NO. B'S TO KEEP --ITYPE=7	0 6440
C	LRY(7,J) = LOCAL ID. NO. OF B'S TO RETAIN.	0 6441
C	LRY(8,J) = LOCAL ID. NO. OF B'S TO RETAIN.	0 6442
C	LRY(9,J) = LOCAL ID. NO. OF B'S TO RETAIN.	0 6443
C		0 6444
C		0 6445
C		0 6446
C	-----CALL READIM (IRY,3,NCYC,3,KR)	0 6447
C		0 6448
C	NOTE-----IRY(1,J) = RCOT TOLERANCE EXPONENT.	0 6449
C	IRY(2,J) = GAIN TOLERANCE EXPONENT	0 6450
C	IRY(3,J) = RCOT TOLERANCE EXPONENT	0 6451
C	USED TO REMOVE SHIFT FREQ.	0 6452
C	IN SUBROUTINE NUMS.	0 6453
C		0 6454
C	DO 500 ICYC = 1,NCYC	0 6455
C		0 6456
C	-----TITLE	0 6457
C	-----LPNAME,LPTAPE,LEIGV,LPLY	0 6458
C		0 6459
C	IF (LEIGV.NE.4HEIGV) GO TO 23	0 6460
C	-----FQMIN,FQMAX	0 6461
C	IMAGINARY PART .GT. FQMIN AND .LT. FQMAX	0 6462
C	23 CONTINUE	0 6463
C		0 6464
C	IF (LPGLY.NE.4HPOLY) GO TO 41	0 6465
C	COMPUTE AND OUTPUT THE TRANSFER FUNCTION IN EXPANDED	0 6466
C	POLYNOMIAL FORMAT	0 6467
C	41 CONTINUE	0 6468
C		0 6469
C	IF (LPTAPE.EQ.4H7IRK) DUMP /PSTUFF/ ON 7-TRACK TAPE (UNIT 12)	0 6470
C	CONTINUE	0 6471
C		0 6472
C	DO 500 ICP = 1,5	0 6473
C		0 6474
C		0 6475
C	IF (LPNAME(ICP) .EQ. 4H) GO TO 500	0 6476
C	IF (LPNAME(ICP) .EQ. 4HBODE	0 6477
C	*.OR. LPNAME(ICP) .EQ. 4HNICH	0 6478
C	*.OR. LPNAME(ICP) .EQ. 4HNYQU	0 6479
C	*.OR. LPNAME(ICP) .EQ. 4HECNN	0 6480
C	*.OR. LPNAME(ICP) .EQ. 4HNINY) GO TO 200	0 6481
C	IF (LPNAME(ICP) .EQ. 4HRCUT) GO TO 300	0 6482
C		0 6483
C	GO TO 9959	0 6484
C		0 6485
C	200 CONTINUE	0 6486
C		0 6487
C	-----JODE, NICHOLS, NYQUIST SECTION-----	0 6488
C		0 6489
C	-----FMIN, FMAX, DBMIN, DBMAX, AMIN, AMAX	0 6490
C		

C	GO TO 300	0 6491
C		0 6492
C	390 CONTINUE	0 6493
C	-----ROOT LOCUS SECTION-----	0 6494
C		0 6495
C	-----CALL READIM (IJM, 2, NRLOC, 2, KR)	0 6496
C	-----CALL READ (RLC, 7, NRLOC, KR, KR)	0 6497
C		0 6498
C	GO 350 IRC =1,NRLOC	0 6499
C		0 6500
C	350 CONTINUE	0 6501
C	GO TO 300	0 6502
C		0 6503
C	400 CONTINUE	0 6504
C		0 6505
C	500 CONTINUE	0 6506
C	RETURN	0 6507
C		0 6508
C	-----DIMENSIONED WORK SPACES-----	0 6509
C		0 6510
C		0 6511
	DIMENSION VS1(414),VS2(207)	412 6512
	DIMENSION LPNAME(5)	0 6513
	DIMENSION GNB(2),GOB(2)	0 6514
	DIMENSION RRN(100),RIN(100),RRD(100),RID(100),R2R(100),R2I(100)	414 6515
	DIMENSION IJM(2,100),LRY(9,100),IRY(3,100),KUKP(3)	418 6516
	DIMENSION RUCTRE(100),ROOTIC(100),W(100,4),IRDW(100,2)	415 6517
	DIMENSION XR(100),XI(100),VR(100),VI(100)	416 6518
		0 6519
C	-----COMMON BLOCKS-----	0 6520
C		0 6521
C		0 6522
	COMMON /KDSIZE/	0 6523
1	KR, KRT, KRX, KVI, KV2, KVA	0 6524
	COMMON /LDSIZE/	0 6525
2	NX, NY, NDLTA, NXSS, NB, NJ1, NY2, ND2	0 6526
	COMMON /LCOUNT/	0 6527
3	NNR, ICN, NNZ, NDR, ICD, NDZ	0 6528
	COMMON /TAPENO/	0 6529
4	NOT1, NOT2, NOT3	0 6530
	COMMON /MISCNO/	0 6531
5	NOPRNT, NOPLET	0 6532
	COMMON /LBDATA/	0 6533
6	FMIN, FMAX, CBMIN, CBMAX	0 6534
	COMMON /LTOL /	0 6535
7	TCLN,TOLD	0 6536
	COMMON /LTITLE/	0 6537
8	TITLE(20)	0 6538
	COMMON /LIJV /	0 6539
9	IV(250) , JV (250)	420 6540
	COMMON /LRARAY/	0 6541
A	FBRN(100), FBNC(100), FBRD(100), FBDC(100)	422 6542
	COMMON /LRDOT /	0 6543
B	R(207) , RX (414)	424 6544
	COMMON / LV1 /	0 6545
C	V1 (100), V2 (100), V3 (100)	426 6546
	COMMON / LV2 /	0 6547
D	XV1 (100), XV2 (100), XV3 (100), XV4 (100)	428 6548
	COMMON /VECTOR/	0 6549
E	Y (250), YD (250)	430 6550

COMMON /LWORK1/	0 6551
F W1(100,100), W2(100,100)	432 6552
COMMON /TIMESS/	0 6553
G ST, DT, T, ET, TMST	0 6554
COMMON /PLTOTA/	0 6555
I NRPLUT, NCPLUT	0 6556
COMMON /PSTUFF/	0 6557
* SAVEO(500), SAVEP(500), SAVED(500), SAVEA(500), KSAVE	447 6558
C	0 6559
C	0 6560
C -----EQUIVALENCE MAP-----	0 6561
C	0 6562
C	0 6563
C	0 6564
EQUIVALENCE (VS1(208),VS2(1))	434 6565
C	0 6566
C	0 6567
C -----DATA STATEMENTS-----	0 6568
C	0 6569
C	0 6570
DATA LBCDE, LROOT, LNYQU, LNICF, LNINY, LTIME, LBLNK, LBONN/	0 6571
* 4HBCDE,4HROOT,4HNYQU,4HNICF,4FNINY,4HTIME,4H ,4HBONN/	0 6572
C	0 6573
DATA NIT/ 5/	0 6574
DATA NUT/ 6 /	0 6575
DATA IFST,NUT12,L7TRK/0,12,4H7TRK/	0 6576
DATA LEGVAL/4HEIGV/	0 6577
DATA LPOLYN/4HPOLY/	0 6578
C	0 6579
LOGICAL DEBUG,LEQU	0 6580
COMMON /LDEBUG/ DEBUG(120)	0 6581
EQUIVALENCE (LEQU,DEBUG(64))	0 6582
C	0 6583
1003 FORMAT (20A4)	0 6584
1004 FORMAT (6F10.0)	0 6585
2001 FORMAT (//,5X,10HON 1CYC = ,12,26H,NUMERATOR GAIN LESS THAN .	0 6586
* D10.4,16HWAS ENCOUNTERED.,//,5X,	0 6587
* 23HTHE BODE GAIN VALUE IS D10.4,//,5X,	0 6588
* 47HPROGRAM CONTINUING WITH NEXT TRANSFER FUNCTION.)	0 6589
C	0 6590
IF(LEQU) WRITE(NUT,1000)	0 6591
1000 FORMAT (' SUBROUTINE DYN540 ')	0 6592
C	0 6593
C -----SET UP FIXED INTEGER DATA-----	0 6594
C	0 6595
KR = 100	436 6596
KRT = 207	438 6597
KRX = 414	440 6598
KV1 = 414	442 6599
KV2 = 207	444 6600
KVX = 100	446 6601
K3 = 3	0 6602
K9 = 9	0 6603
C	0 6604
C -----SET UP VARIABLE INTEGER DATA-----	0 6605
C	0 6606
NY2 = NX - NDLTA - NXSS	0 6607
ND2 = NDLTA - NB	0 6608
C	0 6609
REWIND NUT2	0 6610

REWIND NUT3	0 6611
C	0 6612
C READ IN LINEARIZED PARTIAL DERIVATIVE MATRIX FROM UNIT NUT2	0 6613
C PERFORM SIMILARITY TRANSFORMATION, AND PUT A* BACK ON NUT2.	0 6614
C	0 6615
C FROM SUBROUTINE CONTRL:	0 6616
C NXSS = NUMBER OF SENSOR SIGNALS	0 6617
C NB = NUMBER OF TORQUE SIGNALS	0 6618
C NX, NJQ COMPUTED IN LINEAR	0 6619
C R = ZNEW VECTORS COMPUTED IN LINEAR	0 6620
C W1 = THE H MATRIX OF EQUATION III-21 VOL 1	0 6621
C W2 = THE C MATRIX OF EQUATION III-26 VOL 1	0 6622
C	0 6623
C	0 6624
C NAUX = NXSS + NB	0 6625
C DO 50 L=1,NX	0 6626
C READ (NUT2) (R(I),I=1,NJQ)	0 6627
C DO 45 I=1,NX	0 6628
45 W1(I,L) = R(I)	0 6629
C	0 6630
C DO 46 I=1,NAUX	0 6631
C J = NX+1	0 6632
46 W2(I,L) = R(J)	0 6633
50 CONTINUE	0 6634
C IF (LECU) WRITE (NUT,1001)	0 6635
1001 FORMAT (' -A- IS THE COEFFICIENT MATRIX FOR THE LINEARIZED PERTURB	0 6636
*ATION EQUATIONS ')	0 6637
C CALL WRITE (W1,NX,NX,3H-A-,KR)	0 6638
C IF (LECU) WRITE (NUT,1005)	0 6639
1005 FORMAT (' -A-AUX IS THE COEFFICIENT MATRIX USED TO DEFINE SENSOR A	0 6640
*ND TORQUE SIGNALS AS LINEAR FUNCTIONS OF STATE VARIABLES ')	0 6641
C CALL WRITE (W2,NAUX,NX,4H-A-AUX,KR)	0 6642
C	0 6643
C REWIND NUT2	0 6644
C WRITE (NUT2) ((W1(I,J),I=1,KR),J=1,KR)	0 6645
C WRITE (NUT2) ((W2(I,J),I=1,KR),J=1,KR)	0 6646
C	0 6647
C REWIND NUT2	0 6648
C GET COMPLEX ROOTS OF THE LINEARIZED COUPLED SYSTEM EQUATIONS	0 6649
C PLANT PLUS CONTROL SYSTEM EQUATIONS	0 6650
C CALL QRDHVR (W1,NX,KRQ,RID,KR)	0 6651
C READ (NUT2)((W1(I,J),I=1,KR),J=1,KR)	0 6652
C READ (NUT2) ((W1(I,J),I=1,KR),J=1,KR)	0 6653
C REWIND NUT2	0 6654
C	0 6655
C W1 CONTAINS THE COEFFICIENT MATRIX FOR THE SENSOR AND TORQUE	0 6656
C SIGNALS	0 6657
C CALL ASIMLR (W1,W2,IV,KR)	0 6658
C SAVE W2 = R**(-1)*H*K OF EQUATION III-24 VOL 1 RECORDED FROM	0 6659
C WRITE (NUT2) ((W2(I,J),I=1,KR),J=1,KR)	0 6660
C REWIND NUT2	0 6661
C IF (LECU) WRITE(NUT,1002)	0 6662
1002 FORMAT (' -A*- IS THE COEFFICIENT MATRIX FOR THE LINEARIZED EQUATI	0 6663
*ONS OF MOTION WITH SENSOR AND TORQUE SIGNALS ')	0 6664
C CALL WRITE (W2,NX,NX,4H-A*- ,KR)	0 6665
C GET COMPLEX ROOTS OF TRANSFORMED EQUATIONS	0 6666
C CALL QRDHVR (W2,NX,KRN,RIN,KR)	0 6667
C REMOVE NUMBERS SMALLER THAN 1.0-5 FROM ROOT ARRAYS.	0 6668
C	0 6669
C CALL SIFT (RRD,NX,1.0-5)	0 6670

CALL SIFT (RID,NX,1,D-5)	0 6671
CALL SIFT (RRN,NX,1,D-5)	0 6672
CALL SIFT (RIN,NX,1,D-5)	0 6673
C	0 6674
C PRINT BOTH SETS OF COMPLEX ROOTS, THEY SHOULD BE EQUAL	0 6675
C THEIR CLOSENESS IS A QUALITY MEASURE FOR THE	0 6676
C SIMILARITY TRANSFORMATION	0 6677
CALL RWRITE (2,RRD,RID,RRN,RIN,NX,NX,4HRT A,4HRTA*)	0 6678
READ (NUT2)((W2(I,J),I=1,KR),J=1,KR)	0 6679
REWIND NUT2	0 6680
C	0 6681
C READ IN CONTROL VARIABLE, LNAM AND BRANCH TO APPROPRIATE SECTION.	0 6682
C***** READ IN TYPE OF STABILITY ANALYSIS	0 6683
READ (NIT,1003) LNAM	0 6684
C LNAM = 4HTIME LINEARIZED TIME RESPONSE	0 6685
C = 4HFREQ FREQUENCY RESPONSE ANALYSIS	0 6686
C = BLANK RETURN	0 6687
C	0 6688
IF (LNAM .EQ. LBLNK) RETURN	0 6689
IF (LNAM .EQ. LTIME) GO TO 400	0 6690
C	0 6691
C ----FREQUENCY DOMAIN ANALYSIS SECTION----	0 6692
C	0 6693
C	0 6694
C READ IN FREQUENCY ANALYSIS CONTROL VARIABLES	0 6695
C***** READ SPECIFICATIONS TO DEFINE TRANSFER FUNCTIONS	0 6696
C LRY = ARRAY DEFINES TRANSFER FUNCTION SPECIFICATION DATA	0 6697
C NCYC = NUMBER OF SEPERATE TRANSFER FUNCTIONS TO CONSIDER	0 6698
CALL READIM (LRY,N3,NCYC,K9,KR)	0 6699
	0 6700
	0 6701
C	0 6702
C***** READ EXPONENT TOLERANCE DATA	0 6703
CALL READIM (IRY,N3,NC2,K3,KR)	0 6704
C	0 6705
IF (NC2 .NE. NCYC) GO TO 9999	0 6706
C	0 6707
DO 500 ICYC = 1,NCYC	0 6708
C	0 6709
ITYPE = LRY(1,ICYC)	0 6710
ITFIN = LRY(2,ICYC)	0 6711
JTFOUT = LRY(3,ICYC)	0 6712
KPLOT = LRY(4,ICYC)	0 6713
IAFLG = LRY(5,ICYC)	0 6714
C	0 6715
ITYPE = TRANSFER FUNCTION TYPE	0 6716
C	0 6717
ITFIN = INTEGER TO DEFINE INPUT SIGNAL FOR TRANSFER FUNCTION	0 6718
C	0 6719
JTFOUT = INTEGER TO DEFINE OUTPUT SIGNAL	0 6720
C	0 6721
KPLOT = INTEGER TO DEFINE IF PLOTS REQUIRED	0 6722
C	0 6723
IAFLG = SHOULD CHARACTERISTIC EQUATION OR ITS TRANSPOSE BE USED	0 6724
C	0 6725
TO GET TRANSFER FUNCTION	0 6726
	0 6727
	0 6728
	0 6729
	0 6730
IF (ITYPE .EQ. 0) GO TO 500	
C	
IF (IABS(ITYPE) .LE. 6) GO TO 55	
NBKP = LRY(6,ICYC)	
DO 54 I=1,NBKP	
54 KBKP(I) = LRY(I+6,ICYC)	
55 CONTINUE	
C	
C	
C SET UP DEFAULT VALUES IN ARRAY IRY.	

C		0 6731
	IF (IRY(1,ICYC) .EQ. 0) IRY(1,ICYC) = -7	0 6732
	IF (IRY(2,ICYC) .EQ. 0) IRY(2,ICYC) = -7	0 6733
	IF (IRY(3,ICYC) .EQ. 0) IRY(3,ICYC) = -7	0 6734
C		0 6735
	TCLN = 10.00 ** IRY(1,ICYC)	0 6736
	CTCL = 10.00 ** IRY(2,ICYC)	0 6737
	FTCL = 10.00 ** IRY(3,ICYC)	0 6738
C		0 6739
	TCLD = TCLN	0 6740
C		0 6741
C		0 6742
	IF (IABS(ITYPE) .LT. 1 .OR. ITYPE .GT. 0) GO TO 9999	0 6743
C		0 6744
	C***** READ TITLE FOR TRANSFER FUNCTION ICYC	0 6745
	READ (NIT,1003) TITLE	0 6746
C		0 6747
	READ (NUT2) ((W2(I,J),I=1,KR),J=1,KR)	0 6748
C	W2 = R**(-1)*H*R IN EQUATION III-24 VOL 1	0 6749
	REWIND NUT2	0 6750
C		0 6751
	CALL IFTYPE (W2,W1,V1,NX,NA,ITYPE,ITFIN,NBKP,CBKP,KR,KR)	0 6752
C	EXIT IFTYPE WITH	0 6753
C	W1 = CHARACTERISTIC MATRIX	0 6754
C	V1 = CORRESPONDING INPUT COEFFICIENTS	0 6755
C	ASSOCIATED WITH PARTICULAR TYPE TRANSFER FUNCTION	0 6756
C	PUT CHARACTERISTIC MATRIX ON NUT3	0 6757
	CALL WRITE (W1,NA,NA,4H-AR-,KR)	0 6758
	CALL WRITE (V1,1,NA,4HBCCL,1)	0 6759
	WRITE (NUT3) ((W1(I,J),I=1,KR),J=1,KR)	0 6760
	REWIND NUT3	0 6761
C		0 6762
C	NOW HAVE REDUCED IF A* ON NUT3.	0 6763
C		0 6764
C	OBTAIN ROOTS FOR DENOMINATOR.	0 6765
C		0 6766
C		0 6767
	CALL GCRDVR (W1,NA,KRD,RID,KR)	0 6768
	CALL SIFT (RRD,NA,TOLD)	0 6769
	CALL SIFT (RID,NA,TOLD)	0 6770
C		0 6771
	READ (NUT3) ((W2(J,I),I=1,KR),J=1,KR)	0 6772
	REWIND NUT3	0 6773
C		0 6774
C	GET ROOTS OF TRANSPOSE OF CHARACTERISTIC EQUATION	0 6775
	CALL GCRDVR (W2,NA,V2,V3,KR)	0 6776
	CALL SIFT (V2,NA,TOLD)	0 6777
	CALL SIFT (V3,NA,TOLD)	0 6778
	CALL RWRITE (2,RRD,RID,V2,V3,NA,NA,4H-AR-,4H-RAR)	0 6779
C		0 6780
	IF (IAPLG .EQ. 0) GO TO 59	0 6781
	DO 58 I=1,NA	0 6782
	RRD(I) = V2(I)	0 6783
	58 RID(I) = V3(I)	0 6784
	59 CONTINUE	0 6785
C		0 6786
C	OBTAIN ROOTS OF NUMERATOR.	0 6787
C		0 6788
	READ (NUT3) ((W1(I,J),I=1,KR),J=1,KR)	0 6789
	REWIND NUT3	0 6790

C	LOCATE OUTPUT SIGNAL FOR COMPUTATION OF PARTICULAR TYPE	0 6791
C	TRANSFER FUNCTION CALLED FOR	0 6792
C	OBTAIN ABSOLUTE JTF LOCATION BASED UPON LOCAL JTFOUT.	0 6793
	JTF = NY2 + JTFOUT	0 6794
	IF (IABS(IITYPE) .EQ. 2) JTF = ND2 + JTFOUT	0 6795
	IF (IABS(IITYPE) .EQ. 3) JTF = JTF + NXSS + ND2	0 6796
	IF (IABS(IITYPE) .EQ. 7) JTF = JTF + NXSS + ND2,	0 6797
	IF (IABS(IITYPE) .EQ. 8) JTF = JTFOUT	0 6798
	CALL NQMS (W1,W2,V1,RRN,RIN,R2R,R2I,PTOL,	0 6799
	* GAIN,IFLG,NNUM,NZRO,JTF,NA,KR)	0 6800
C		0 6801
	NN = NZRO	0 6802
	ND = NA	0 6803
C		0 6804
	IF (IFLG .NE. 0) GO TO 65	0 6805
64	IFLG = 0	0 6806
	CALL PAGEHD	0 6807
	WRITE (NCT,2001) ICYC,GTOL,GB	0 6808
	GO TO 75	0 6809
65	CONTINUE	0 6810
C		0 6811
	CALL SIFT (RRN,NN,IOLN)	0 6812
	CALL SIFT (RIN,NN,IOLN)	0 6813
C	CALL DCQRT TO SEPERATE NUMERATOR OR ROOTS INTO REAL AND ZERO	0 6814
C	ROOTS (RETURNED IN V1) AND COMPLEX PAIRS (RETURNED IN V2)	0 6815
	CALL DCQRT (RRN,RIN,NN,NNR,ICN,NNZ,V1,V2)	0 6816
	CALL DFCRMB (NNR,ICN,V1,V2,FBRN,FBNC,V3,1.00,GAIN,GNB)	0 6817
C		0 6818
70	CONTINUE	0 6819
C		0 6820
	CALL RWRITE (2,RRN,RIN,RRD,RID,NN,ND,4H NUM,4H DEN)	0 6821
C		0 6822
C	SEPERATE OR ROOTS OF DENOMINATOR	0 6823
	CALL DCQRT (RRD,RID,ND,NDR,ICD,NDZ,V1,V2)	0 6824
	CALL DFCRMB (NDR,ICD,V1,V2,FBRD,FBDC,V3,1.00,1.00,GDB)	0 6825
	GB = GNB(2)/GDB(2)	0 6826
C	GE = EODE GAIN SEE EQUATION III-48 AND III-70. VOL 1	0 6827
C	ARRAYS FBRN, FBNC, FBRD, AND FBDC CONTAIN SYSTEM TIME CONSTANTS	0 6828
C	DAMPING AND FREQUENCIES REQUIRED IN EQUATION III-70. VOL 1	0 6829
	IF (CABS(GB) .LT. GTOL) GO TO 64	0 6830
C		0 6831
C	-----ESTABLISH WORKING ROOT COUNTS.	0 6832
C		0 6833
	KNR = NNR	0 6834
	KDR = NDR	0 6835
	KNZ = NNZ	0 6836
	KDZ = NDZ	0 6837
	KCN = ICN	0 6838
	KCD = ICD	0 6839
C		0 6840
	CALL ZERC (XV1,1,KVX,1)	0 6841
	CALL ZERC (XV2,1,KVX,1)	0 6842
	CALL ZERC (XV3,1,KVX,1)	0 6843
	CALL ZLRC (XV4,1,KVX,1)	0 6844
C		0 6845
	IF (KNR .EQ. 0) GO TO 2202	0 6846
	DO 201 I=1,KNR	0 6847
201	XV1(I) = FBRN(I)	0 6848
2202	CONTINUE	0 6849
	IF (KCR .EQ. 0) GO TO 2203	0 6850

DO 202 I=1,KDR	0 6851
202 XV3(I) = FBRD(I)	0 6852
2203 CONTINUE	0 6853
IF (KCN .EQ. 0) GO TO 2204	0 6854
K=2*KCN	0 6855
DO 203 I=1,K	0 6856
203 XV2(I) = FUNC(I)	0 6857
2204 CONTINUE	0 6858
IF (KCD .EQ. 0) GO TO 205	0 6859
K=2*KCD	0 6860
DO 204 I=1,K	0 6861
204 XV4(I) = FUNC(I)	0 6862
205 CONTINUE	0 6863
C	0 6864
C-----REMOVE REAL ZEROS PRIOR TO CALL TO TTF	0 6865
IF (KNR .NE. 0) CALL RMVZRU (XV1,KNR)	0 6866
IF (KDR .NE. 0) CALL RMVZRU (XV3,KDR)	0 6867
CALL ZLHC (R,1,KRT,1)	0 6868
C	0 6869
C CALL TTF TO SET UP THE ROOTS ARRAY R	0 6870
CALL TTF (KNR,KCN,KNZ,KDR,KCD,KDZ,	0 6871
* GB,XV1,XV2,XV3,XV4,R,KRT)	0 6872
C	0 6873
C CALL CANCEL TO CANCEL OUT ROOTS COMMON TO NUM. AND DEN.	0 6874
CALL CANCEL (R)	0 6875
IF(.NOT.LEQU) GO TO 10	0 6876
WRITE(NIT,1006)	0 6877
1006 FORMAT (' ROOT ARRAY OUT OF CANCEL IS ')	0 6878
CALL WRITE5(R,1,KRT,1)	0 6879
10 CONTINUE	0 6880
C	0 6881
C	0 6882
C***** READ IN DISPLAY CONTROL VARIABLES	0 6883
C	0 6884
75 READ(NIT,1003) LPNAME,LPTAPE,LEIGV,LPDLY	0 6885
C	0 6886
C CHECK TO SEE IF EIGENVECTORS ARE TO BE COMPUTED	0 6887
C READ FREQUENCY SPREAD FOR ROOTS OF CHARACTERISTIC EQUATION	0 6888
IF(LEIGV.NE.LEGVAL) GO TO 23	0 6889
C	0 6890
C*****READ IN RANGE FOR EIGENVECTOR COMPUTATION	0 6891
C	0 6892
READ(NIT,1004) FQMIN,FQMAX	0 6893
IF(IFLS.EQ.0) GO TO 20	0 6894
IC = 0	0 6895
DO 24 I=1,ICD	0 6896
IF (V2(2*I).LT.FQMIN.OR.V2(2*I).GT.FQMAX) GO TO 24	0 6897
IC = IC+1	0 6898
ROOTIE(IC) = V2(2*I-1)	0 6899
ROOTIE(IC) = V2(2*I)	0 6900
24 CONTINUE	0 6901
C	0 6902
C	0 6903
C PUT ROOT IN ORDER	0 6904
DO 26 M=1,IC	0 6905
VSV = ROOTIE(M)	0 6906
ISV = M	0 6907
M1 = M+1	0 6908
IF(M1.LE.IC) GO TO 26	0 6909
DO 27 N=M1,IC	0 6910
IF (VSV.GE.ROOTIE(N)) GO TO 27	

VSV = ROOTIE(N)	0 6911
ISV = N	0 6912
27 CONTINUE	0 6913
28 CONTINUE	0 6914
WSV = ROOTRE(ISV)	0 6915
ROOTIE(ISV) = ROOTIE(M)	0 6916
ROOTRE(ISV) = ROOTRL(M)	0 6917
ROOTIE(M) = VSV	0 6918
ROOTRE(M) = WSV	0 6919
26 CONTINUE	0 6920
C	0 6921
CALL WRITE(ROOTRE,1,IC,6HROOTRE,1)	0 6922
CALL WRITE(ROOTIE,1,IC,6HROOTIE,1)	0 6923
CALL PAGEHD	0 6924
WRITE(NCT,1010)	0 6925
1010 FORMAT (///,10X,'REFER TO THE TRANSFORMED STATE VECTOR CORRELATION	0 6926
* ARRAY AND CODE ITYPE NUMBERING LOGIC IN SUBROUTINE IFTYPE',/,	0 6927
* 10X,'TO ASSOCIATE EACH EIGENVECTOR ELEMENT WITH A PARTICULAR DEGR	0 6928
*EE OF FREEDOM')	0 6929
C	0 6930
DO 25 M=1,IC	0 6931
I = IC+1-M	0 6932
READ (NCT3) ((W1(I1,J),I1=1,KR),J=1,KR)	0 6933
REWIND NCT3	0 6934
CALL ASQR(0,W1,W2,NA,KR)	0 6935
CALL EIGVEC(3,W1,W2,W1,KRW,XR,XI,VR,VI,ROOTRE(I),ROOTIE(I),NA,KR,0	0 6936
*SW1,COUNT,ERR)	0 6937
WRITE (NCT,1007) ROOTRE(I),ROOTIE(I)	0 6938
1007 FORMAT (///,10X,'EIGENVALUE = ',D15.8,' + I* ',D15.8, '//)	0 6939
WRITE (NCT,1008)	0 6940
1008 FORMAT (10X,'EIGENVECTOR OF CHARACTERISTIC MATRIX -AR-',16X,	0 6941
* 'EIGENVECTOR OF TRANSPOSE OF CHARACTERISTIC MATRIX -AR**T-',//)	0 6942
WRITE(NCT,1009) (J,XR(J),XI(J),J,VR(J),VI(J),J=1,NA)	0 6943
1009 FORMAT (5X,15,10X,D15.8,' + I*(',D15.8,')',	0 6944
* 5X,15,10X,D15.8,' + I*(',D15.8,')')	0 6945
WRITE(NCT,1011)	0 6946
1011 FORMAT (/,10X,'SAME EIGENVECTOR EXPRESSED IN POLAR COORDINATES WIT	0 6947
*H PHASE GIVEN IN DEGREES (MAG,PHASE)',//)	0 6948
DO 11 J=1,NA	0 6949
XMG = DSQRT(XR(J)**2 + XI(J)**2)	0 6950
VMG = DSQRT(VR(J)**2 + VI(J)**2)	0 6951
IF(XMG.EQ.0.000) GO TO 12	0 6952
XAG = DATAN2(XI(J),XR(J))*57.29577950+00	0 6953
XR(J) = XMG	0 6954
XI(J) = XAG	0 6955
12 IF(VMG.EQ.0.000) GO TO 11	0 6956
VAG = DATAN2(VI(J),VR(J))*57.29577950+00	0 6957
VR(J) = VMG	0 6958
VI(J) = VAG	0 6959
11 CONTINUE	0 6960
WRITE(NCT,1012) (J,XR(J),XI(J),J,VR(J),VI(J),J=1,NA)	0 6961
1012 FORMAT (5X,15,10X,'(',D15.8,' , ',D15.8,')',	0 6962
* 7X,15,10X,'(',D15.8,' , ',D15.8,')')	0 6963
25 CONTINUE	0 6964
C	0 6965
23 CONTINUE	0 6966
C	0 6967
CHECK TO SEE IF 7 TRACK TAPE OF /PSTUFF/ IS TO BE WRITTEN	0 6968
IF(LPTAPE.NE.L7TRK) GO TO 20	0 6969
IF(IFST.NE.0) GO TO 20	0 6970
IFST = 1	

REXING ACT12	0 6971
20 CONTINUE	0 6972
C	0 6973
DO 500 ICP = 1,5	0 6974
C	0 6975
IF (LPNAME(ICP) .EQ. LBLNK) GO TO 500	0 6976
IF (LPNAME(ICP) .EQ. LBODE	0 6977
*.OR. LPNAME(ICP) .EQ. LNBCH	0 6978
*.OR. LPNAME(ICP) .EQ. LNYDU	0 6979
*.OR. LPNAME(ICP) .EQ. LBCNN	0 6980
*.OR. LPNAME(ICP) .EQ. LNNY) GO TO 200	0 6981
IF (LPNAME(ICP) .EQ. LRCOT) GO TO 300	0 6982
	NEPROR = 2 0 6983
GO TO 9999	0 6984
C	0 6985
200 CONTINUE	0 6986
C	0 6987
-----FREQUENCY RESPONSE PROCESSING-----	0 6988
C	0 6989
C***** READ IN FREQUENCY RESPONSE BANDS FOR COMPUTATION	0 6990
READ (NIT,1004) FMIN, FMAX, DBMIN, DBMAX, AMIN, AMAX	0 6991
IF (IFLG .EQ. 0) GO TO 500	0 6992
C	0 6993
KNR = R(1) + 0.100	0 6994
KCN = R(2) + 0.100	0 6995
KNZ = R(3) + 0.100	0 6996
KDR = R(4) + 0.100	0 6997
KCD = R(5) + 0.100	0 6998
KDZ = R(6) + 0.100	0 6999
CALL WRITE (R,1,KRT,4HRKED,1)	0 7000
C	0 7001
CALL ZERL (XV1,1,KVX,1)	0 7002
CALL ZERL (XV2,1,KVX,1)	0 7003
CALL ZERC (XV3,1,KVX,1)	0 7004
CALL ZLRC (XV4,1,KVX,1)	0 7005
C	0 7006
C LOAD ROOT DATA INTO	0 7007
C XV1 = NUMERATOR REAL	0 7008
C XV2 = NUMERATOR COMPLEX	0 7009
C XV3 = DENOMINATOR REAL	0 7010
C XV4 = DENOMINATOR COMPLEX	0 7011
IF (KNR .EQ. 0) GO TO 2207	0 7012
DO 206 I=1,KNR	0 7013
L=7+I	0 7014
206 XV1(I) = R(L)	0 7015
2207 CONTINUE	0 7016
IF (KCN .EQ. 0) GO TO 2208	0 7017
K=2*KCN	0 7018
DO 207 I=1,K	0 7019
L=7+KNR+I	0 7020
207 XV2(I) = R(L)	0 7021
2208 CONTINUE	0 7022
IF (KDR .EQ. 0) GO TO 2209	0 7023
DO 208 I=1,KDR	0 7024
L = 7+KNR+2*KCN+I	0 7025
208 XV3(I) = R(L)	0 7026
2209 CONTINUE	0 7027
IF (KCD .EQ. 0) GO TO 2210	0 7028
K=2*KCD	0 7029
DO 209 I=1,K	0 7030

L=7+KNR+2*KCN+KDR+1	0 7031
205 XV4(I) = R(L)	0 7032
2210 CONTINUE	0 7033
C	0 7034
C-----EXTEND REAL ARRAY COUNTS TO INCLUDE REAL ZEROS.	0 7035
KNR = KNR + KNZ	0 7036
KDR = KDR + KDZ	0 7037
C	0 7038
C-----PERFORM THE FREQUENCY RESPONSE.	0 7039
C	0 7040
CALL SFREQ2 (KNR,KCN,KDR,KCJ,GB,	0 7041
1 XV1,XV2,XV3,XV4,FMIN,FMAX,TITLE)	0 7042
C	0 7043
C CFLCK TO SEE IF REQUEST MADE TO COMPUTE POLYNOMIAL FORM	0 7044
C OF TRANSFER FUNCTION FOR OUTPUT	0 7045
IF (LFOLY.NE.LPOLYN) GO TO 41	0 7046
CALL PAGEHD	0 7047
CALL RILF(R,VS1,VS2,KRT)	0 7048
WRITE(NCT,38)	0 7049
38 FORMAT('IP(S) =')	0 7050
NP1 = VS1(1)	0 7051
WRITE(NCT,40) VS1(J),(VS1(I+J),I,I=1,NP1)	0 7052
WRITE(NCT,39)	0 7053
39 FORMAT('OQ(S) =')	0 7054
NQ1 = VS1(2)	0 7055
WRITE(NCT,40) VS1(NP1+4),(VS1(I+NP1+4),I,I=1,NQ1)	0 7056
40 FORMAT (3X,D22.15,7X,3(1X,'+(',D22.15,'*S**',12,')')/,	0 7057
* 4(1X,'+(',D22.15,'*S**',12,')'))	0 7058
41 CONTINUE	0 7059
C	0 7060
C	0 7061
IF (LPTAPE.NE.L7TRK) GO TO 21	0 7062
WRITE(NCT12,100) IFST,KSAVE	0 7063
WRITE (NCT12,101) (SAVEG(I),SAVEP(I),SAVE)(I),SAVEA(I),I=1,KSAVE)	0 7064
WRITE(NCT,103)	0 7065
WRITE(NCT,102) TITLE,LPNAME(IOP),IFST	0 7066
IFST = IFST+1	0 7067
100 FORMAT (215)	0 7068
101 FORMAT (4E18.8)	0 7069
102 FORMAT (20A4.7,3X,A4,' IN FILE',I4,' OF DATA SET 12')	0 7070
103 FORMAT (//,' FREQUENCY RESPONSE DATA FOR')	0 7071
104 FORMAT (//,' ROOT LOCUS DATA FOR ')	0 7072
21 CONTINUE	0 7073
C	0 7074
IF (KPLCT.EQ. 0) GO TO 220	0 7075
IF (LPNAME(IOP).EQ. LBODE	0 7076
* .OR.LPNAME(IOP).EQ. LBONN)	0 7077
* CALL SPLOT (TITLE,FMAX,FMIN,DBMIN,DBMAX)	0 7078
C	0 7079
IF (LPNAME(IOP).EQ. LNICH	0 7080
* .CR.LPNAME(IOP).EQ. LNINY	0 7081
*.OR.LPNAME(IOP).EQ. LBONN) CALL NIPLT (TITLE,DBMIN,DBMAX)	0 7082
C	0 7083
IF (LPNAME(IOP).EQ. LNYOU	0 7084
* .OR.LPNAME(IOP).EQ. LBONN	0 7085
* .CR.LPNAME(IOP).EQ. LNINY) CALL NYPLT (TITLE,AMIN,AMAX)	0 7086
C	0 7087
GO TO 500	0 7088
220 CALL PAGEHD	0 7089
WRITE (NCT,221) ICYC, LPNAME(IOP)	0 7090

221 FORMAT (//,10X,	0 7091
* 48FNO FREQUENCY RESPONSE PLOTS GENERATED ON ICYC = 13,	0 7092
* //,10X,9HLPNAME = A4)	0 7093
GO TO 500	0 7094
300 CONTINUE	0 7095
C	0 7096
C	0 7097
C	0 7098
C	0 7099
CALL RTCP (R,RX,VS1,KRX)	0 7100
NP = RX(1) + 1.00	0 7101
NQ = RX(2) + 1.00	0 7102
C	0 7103
GO 320 I=1,NP	0 7104
J=I+2	0 7105
320 XV1(I) = RX(J)	0 7106
GO 325 I=1,NQ	0 7107
J = I+2+NP	0 7108
325 XV2(I) = RX(J)	0 7109
CALL WRITE (XV2,1,NQ,4HPDEN,1)	0 7110
CALL WRITE (XV1,1,NP,4HPNUM,1)	0 7111
C	0 7112
C	0 7113
C	0 7114
C	0 7115
C	0 7116
***** READ IN ROOT LOCUS CONTROL DATA	0 7117
330 CALL READIM (IJM,NR2,NRLC,2,KR)	0 7118
IF (IFLG .EQ. 0) GO TO 340	0 7119
IF (NR2 .NE. 2 .OR. NRLC .GT. KR) GO TO 999	0 7120
C	0 7121
C-----NOTE..... IJM(1,J) = ISNM(J)	0 7122
C	0 7123
C	0 7124
C	0 7125
C	0 7126
***** READ IN ADDITIONAL ROOT LOCUS DATA	0 7127
340 CALL READ (W1,NR2,NC2,KR,KR)	0 7128
IF (IFLG .EQ. 0) GO TO 500	0 7129
IF (NR2 .NE. 6 .OR. NC2 .NE. NRLC) GO TO 999	0 7130
C	0 7131
C-----NOTE.....W1(1,J) = THETAC(J)	0 7132
C	0 7133
C	0 7134
C	0 7135
C	0 7136
C	0 7137
C	0 7138
C	0 7139
C	0 7140
C	0 7141
C	0 7142
C	0 7143
C	0 7144
C	0 7145
C	0 7146
C	0 7147
C	0 7148
C	0 7149
C	0 7150

C		C 7151
C	LOCATE PROPER STARTING ROOT.	0 7152
C		0 7153
C	IF (ISNIM .NE. 1) GO TO 341	0 7154
C		0 7155
C	ROOT IS AN OPEN LOOP ZERO.	0 7156
C		0 7157
	SR = RRN(JJ)	0 7158
	SI = RIN(JJ)	0 7159
	GO TO 342	0 7160
341	CONTINUE	0 7161
C		0 7162
C	ROOT IS A POLE.	0 7163
C		0 7164
	SR = RRN(JJ)	0 7165
	SI = RIN(JJ)	0 7166
342	CONTINUE	0 7167
C		0 7168
	CALL RLOCUS (XV1,XV2,SCL,SR,SI,NP,NQ,THETAJ,	0 7169
	1 XMIN,XMAX,YMAX,ALOC)	0 7170
C		0 7171
	IF(LFTAPE.NE.L7TRK) GO TO 22	0 7172
	WRITE(NOT12,100) IFST,KSAVE	0 7173
	WRITE(NOT12,101) (SAVED(I),SAVEP(I),SAVED(I),SAVEA(I),I=1,KSAVE)	0 7174
	WRITE(NOT,104)	0 7175
	WRITE(NOT,102) TITLE,LPNAME(IOP),IFST	0 7176
	IFST = IFST+1	0 7177
22	CONTINUE	0 7178
C		0 7179
	IF (KPLUT .EQ. 1) CALL RLPLUT (TITLE,ISNIM,ICYC,IRC)	0 7180
C		0 7181
350	CONTINUE	0 7182
C		0 7183
C		0 7184
	GO TO 500	0 7185
400	CONTINUE	0 7186
C		0 7187
C	-----LINEARIZED TIME RESPONSE SECTION-----	0 7188
C		0 7189
	READ (NOT2) ((W1(I,J),I=1,KR),J=1,KR)	0 7190
	REWIND NOT2	0 7191
	CALL LTRESP	0 7192
	RETURN	0 7193
C		0 7194
500	CONTINUE	0 7195
C		0 7196
C		0 7197
	RETURN	0 7198
C		0 7199
	9999 WRITE (NOT,1999) NERROR	0 7200
	1999 FORMAT (1H1, //10X, 44HERROR ENCOUNTERED IN OVERLAY(4,0), NERROR = ,	0 7201
	* 13, //10X, 16HPROGRAM STOPPED.)	0 7202
	STCP	0 7203
C		0 7204
	END	0 7205

```

C      SUBROUTINE ASIMLR (A,B,IV,KR)                                0 7206
C                                                                    C 7207
C      IMPLICIT REAL*8(A-H,O-Z)                                    0 7208
C                                                                    C 7209
C      ---LATEST MOD ON 09/04/75                                    0 7210
C                                                                    C 7211
C      SUBROUTINE ESTABLISHES TRANSFORMED PARTIAL DERIVATIVE MATRIX 0 7212
C      BY PERFORMING A SIMILARITY TRANSFORMATION TO                0 7213
C      EXCHANGE PLANT STATE VARIABLES ,Y, FOR SENSOR SIGNALS,XSS 0 7214
C      AND CONTROL SYSTEM VARIABLES ,DELTA, FOR TORQUE              0 7215
C      VARIABLES ,B.                                                0 7216
C                                                                    0 7217
C      THE VARIABLE SEQUENCE IS REORDERED FROM Y,DELTA,XSS,B      0 7218
C      TO Y,XSS,DELTA,B                                            0 7219
C                                                                    0 7220
C      ----SUBROUTINE ARGUMENT DESCRIPTIONS-----                 0 7221
C                                                                    0 7222
C      A      = INPUT MATRIX OF PARTIAL DERIVATIVES. COORDINATE ORDER 0 7223
C      IS XSS, B. SIZE IS (NJQ-NX),NX.                             0 7224
C      B      = OUTPUT TRANSFORMED AND REORDERED PARTIAL DERIVATIVE 0 7225
C      MATRIX. ORDER IS Y,XSS,DELTA,B. SIZE IS NX,NX.             0 7226
C      IV     = INPUT INTEGER WORK VECTOR. SIZE MUST BE AT LEAST NX. 0 7227
C      KR     = INPUT ROW DIMENSION SIZE OF A AND B IN CALLING PROGRAM. 0 7228
C                                                                    0 7229
C      DIMENSION A(KR,1), B(KR,1), IV(1)                          0 7230
C                                                                    0 7231
C      COMMON /LDSIZE/                                             0 7232
C      2          NX, NY, NDLTA, NXSS, NBTQ, NJQ, NY2, NO2          0 7233
C      COMMON /SPECIF/                                             0 7234
C      *      BETAH(6, 6),BETAND(6, 6),AMO(2, 5),RH(3,3,30),KS(3,3,30), 16 7235
C      *      DH(3,35),DS(3,30),IMD(3, 5),NMOX(6, 5),LFTSMX(15), 17 7236
C      *      NB,NH,NBPT,NCFMC,NDELTA,ITCPOL(2, 6),IRGFLX( 6),IHDATA(7, 6), 18 7237
C      *      LCU(14),LENU(14),NU,NBETA,NLAM,NEQ 19 7238
C      COMMON /IAPEND/                                             0 7239
C      4          NOT1, NOT2, NOT3 0 7240
C      COMMON /VECTOR/                                             0 7241
C      E          Y (250), YD (250) 430 7242
C      COMMON /VINDEP/                                             0 7243
C      *      INDEP(250) 21 7244
C      DATA NOT/0/ 0 7245
C      LOGICAL DEBUG,LEQU 0 7246
C      COMMON /LDEBUG/ DEBUG(120) 0 7247
C      EQUIVALENCE (LEQU,DEBUG(65)) 0 7248
C                                                                    0 7249
C      SET OF C. LOWER PART OF -A- REQUIRED TO OBTAIN -I-.          0 7250
C                                                                    0 7251
C      NR = NJQ - NX 0 7252
C      NC = NJQ 0 7253
C      KC = (3*KR)/5 0 7254
C      IF(LEQU) WRITE(NOT,1000) 0 7255
C 1000 FORMAT (' SUBROUTINE ASIMLR ') 0 7256
C      IF(,NOT,LEQU) GO TO 50 0 7257
C      WRITE (NOT,1001) 0 7258
C 1001 FORMAT (' INPUT MATRIX COEFFICIENTS FOR LINEARIZED SIGNAL EQUATION 0 7259
C      *S ') 0 7260
C      CALL WRITES(A,NR,NX,KR) 0 7261
C      50 CONTINUE 0 7262
C      THE USE OF KC PERMITS THE B SPACE TO BE BETTER UTILIZED 0 7263
C      VIA EXTENDING THE PERMISSABLE NC. OF AUXILIARY EQUATIONS. 0 7264
C                                                                    0 7265

```


CALL ZERC (B,KR,KR,KR)	0 7266
C SET UP B SPACE FOR KC ROW DIMENSION WITH AUGMENTED A.	0 7267
C	0 7268
DO 1C I=1,NR	0 7269
L = I+NX	0 7270
DO 1C J=1,NX	0 7271
C	0 7272
IJ2 = I+KC*(J-1)	0 7273
J2 = (IJ2-1)/KR+1	0 7274
I2 = IJ2-KR*(J2-1)	0 7275
C	0 7276
E(I2,J2) = A(I,J)	0 7277
C	0 7278
IF (I .NE. J) GO TO 1D	0 7279
C	0 7280
IJL2 = I+KC*(L-1)	0 7281
JL2 = (IJL2-1)/KR+1	0 7282
IL2 = IJL2-KR*(JL2-1)	0 7283
C	0 7284
E(IL2,JL2) = -1.00	0 7285
C	0 7286
1C CONTINUE	0 7287
C	0 7288
C THE ARRAY B NOW CONTAINS EQUATION III-27 VOL 1	0 7289
C ESTABLISH SEARCH LIMIT FOR SUBROUTINE FINDT	0 7290
C	0 7291
NS = NX	0 7292
CALL FINDT (B,NR,NC,NS,A,NRET,KC,KR)	0 7293
C EXIT FINDT WITH THE SIMILARITY TRANSFORMATION MATRIX R	0 7294
C OF EQUATION III-23 VOL 1, STORED IN A ARRAY	0 7295
CALL WRITE (A,NRET,NRET,4H-T- ,KR)	0 7296
C	0 7297
C A = -T-	0 7298
C	0 7299
C FORM -A*- = T(INV) A I	0 7300
C	0 7301
C READ PARTIAL DERIVATIVE MATRIX H	0 7302
READ (NUT2) ((B(I,J),I=1,KR),J=1,KR)	0 7303
C PLT SIMILARITY TRANSFORMATION MATRIX R ON NUT2 AFTER H	0 7304
WRITE (NUT2) ((A(I,J), I=1,KR),J=1,KR)	0 7305
REWIND NUT2	0 7306
C INVERT -T- USING GAUSSI (THE SIMILARITY TRANSFORMATION MATRIX R)	0 7307
CALL GAUSSI (A,B,NRET,KR)	0 7308
C B = T(INV)	0 7309
C TRANSFORM STATE VECTOR FOR POSSIBLE USE IN LINEARIZED RESPONSE.	0 7310
K = 0	0 7311
DO 15 I=1,NEQ	0 7312
IF (INDLF(I) .EQ. 0) GO TO 15	0 7313
K=K+1	0 7314
Y(K) = Y(I)	0 7315
15 CONTINUE	0 7316
C FIND 2 VECTORS IN EQUATION III-23 VOL 1	0 7317
CALL MULTB (B,Y,NRET,NRET,1,KR,KR)	0 7318
CALL WRITE (Y,1,NRET,4H Y* ,1)	0 7319
C READ PARTIAL DERIVATIVE MATRIX H	0 7320
READ (NUT2) ((A(I,J),I=1,KR),J=1,KR)	0 7321
C FORM R**(-1)*H	0 7322
CALL MULTA (B,A,NX,NX,NX,KR,KR)	0 7323
C B= T(INV) * -A-	0 7324
C READ R	0 7325

READ (NUT2) ((A(I,J),I=1,KR),J=1,KR)	0 7326
REWIND NUT2	0 7327
C FORM R**(-1)*H*R EQUATION 111-24 PUT RESULTS IN A ARRAY	0 7328
CALL MULTB (B,A,NX,NX,NX,KR,KR)	0 7329
IF(.NOT.LEQU) GO TO 150	0 7330
WRITE (NCT,1006)	0 7331
1006 FORMAT (' R**(-1)*H*R MATRIX IS ')	0 7332
CALL WRITES (A,NX,NX,KR)	0 7333
150 CONTINUE	0 7334
C B = -A*-	0 7335
C	0 7336
C RECFOR FROM Y,DELTA,XSS,B	0 7337
C TO Y,XSS,DELTA,B	0 7338
C NY2 = NX - NDLTA - NXSS	0 7339
C ND2 = NDLTA - NDTO	0 7340
C	0 7341
DO 20 I=1,NX	0 7342
20 IV(I) = I	0 7343
DO 30 I=1,NXSS	0 7344
L = NY2 + I	0 7345
K = L + ND2	0 7346
30 IV(K) = L	0 7347
IF(ND2.EQ.0) GO TO 41	0 7348
DO 40 I=1,ND2	0 7349
K = NY2 + I	0 7350
L = NY2 + NXSS + I	0 7351
40 IV(K) = L	0 7352
41 CONTINUE	0 7353
CALL ZERU (B,NX,NX,KR)	0 7354
CALL REVAUD (1.00,A,IV,IV,B,NX,NX,NX,KR,KR)	0 7355
IF(.NOT.LEQU) GO TO 100	0 7356
WRITE(NCT,1007)	0 7357
1007 FORMAT (' REORDERED R**(-1)*H*R MATRIX IS ')	0 7358
CALL WRITES(B,NX,NX,KR)	0 7359
100 CONTINUE	0 7360
C	0 7361
RETURN	0 7362
END	0 7363

SUBROUTINE FINDT (C,NCN,NX,NS,T,NRET,KC,KF)	0 7364
C	0 7365
IMPLICIT REAL*8(A-H,O-Z)	0 7366
C	0 7367
C ---LATEST MOD ON 05/04/75	0 7368
C	0 7369
DIMENSION C(KC,1), T(KT,1)	0 7370
DIMENSION IVEC(207), JVEC(207)	448 7371
C	0 7372
COMMON /LDSIZE/	0 7373
2 NNX, NY, NDLTA, NXSS, NB10, NJ0, NY2, ND2	0 7374
COMMON /SPECIF/	0 7375
* BETAH(6,6),BETAHD(6,6),AMD(2,5),RH(3,3,30),RS(3,3,30),	16 7376
* OH(3,35),DS(3,30),IMU(3,5),NMOW(6,5),IFISM(15),	17 7377
* NB,NH,NSPT,NDFMC,NDLTA,ITCPOL(2,6),IRGFLEX(6),IHDATA(7,6),	18 7378
* LUCCL(14),LENU(14),NU,NBETA,NLAM,NEQ	19 7379

COMMON /LIJV /	0 7380
9 IV(250) , JV (250)	420 7381
COMMON /VINDEP/	0 7382
* INDEP(250)	21 7383
LOGICAL DEBUG,LEQU	0 7384
COMMON /LDEBUG/ DEBUG(120)	0 7385
EQUIVALENCE (LEQU,DEBUG(66))	0 7386
C	0 7387
DATA EPS , NOT / 1.0-15, 6 /	0 7388
1001 FORMAT (/// 5X,36HSUBROUTINE FIND TERMINATED. SING. AT 15)	0 7389
1002 FORMAT (///1X,	0 7390
* 55HTHE TRANSFORMED STATE VECTOR CORRELATION ARRAY FOLLOWS. /1X,	0 7391
* 43HELEMENTS REFER TO ORIGINAL STATE VARIABLES.,/1X,115(1H-))	0 7392
2001 FORMAT (///1X,47HTHE FOLLOWING INTEGER ARRAY (INDEP) PRESCRIBES	0 7393
*54HINDEPENDENT VARIABLES (1), AND DEPENDENT VARIABLES (0). /1X,	0 7394
* 115(1H-))	0 7395
2010 FORMAT (//1X44HTHE STATE VECTOR LENGTH ARRAY (LENU) FOLLOWS ,	0 7396
* /1X,115(1H-))	0 7397
2011 FORMAT (//1X46HTHE STATE VECTOR LOCATION ARRAY (LDCU) FOLLOWS,	0 7398
* /1X,115(1H-))	0 7399
C	0 7400
IF(.NOT.LEQU) GO TO 120	0 7401
WRITE(NCT,1003)	0 7402
1003 FORMAT (' SUBROUTINE FINDT INPUT MATRIX ')	0 7403
(CALL WRITES (C,NCN,NX,KC)	0 7404
120 CONTINUE	0 7405
DO 5 I=1,NX	0 7406
5 JVEC(I) = 1	0 7407
C	0 7408
C FOR PROCEDURE SEE VOL 1,PAGE 63 'ILLUSTRATIVE EXAMPLE'	0 7409
DO 10 L=1,NCN	0 7410
JBIG = 1	0 7411
A = DABS(C(L,1))	0 7412
C FIND PIVOT ELEMENT FOR ROW L	0 7413
DO 15 J=2,NS	0 7414
AT = DABS(C(L,J))	0 7415
IF (AT .LT. A) GO TO 15	0 7416
A = AT	0 7417
JBIG = J	0 7418
15 CONTINUE	0 7419
C PIVOT ELEMENT FOR ROW L IS (L,JBIG)	0 7420
IVC(L) = -JBIG	0 7421
JVEC(JBIG) = 0	0 7422
IF (A .GT. EPS) GO TO 20	0 7423
WRITE (NCT,1001) L	0 7424
STOP	0 7425
20 CONTINUE	0 7426
CLJBIG = C(L,JBIG)	0 7427
DO 17 J=1,NX	0 7428
17 C(L,J) = C(L,J)/CLJBIG	0 7429
DO 25 I=1,NCN	0 7430
F = C(I,JBIG)	0 7431
IF (I .EQ. L) GO TO 23	0 7432
DO 30 J=1,NX	0 7433
30 C(I,J) = C(I,J) - F*C(L,J)	0 7434
25 CONTINUE	0 7435
10 CONTINUE	0 7436
C LOWER CASE D SUB TQ MATRIX ON PAGE 64 FOUND	0 7437
IF(.NOT.LEQU) GO TO 130	0 7438
WRITE(NCT,1004)	0 7439

1300 FORMAT (' VARIABLE EXCHANGE MATRIX IS ')	C 7440
CALL WRITES(C,NCN,IX,KC)	C 7441
130 CONTINUE	C 7442
C	C 7443
NVAL = 0	C 7444
DO 40 I=1,NX	C 7445
IF (JVEC(I) .EQ. 0) GO TO 40	C 7446
NVAL = NVAL + 1	C 7447
JVEC(I) = NVAL	C 7448
40 CONTINUE	C 7449
C	C 7450
CONSTRUCT NRET X NRET SIMILARITY TRANSFORMATION MATRIX	C 7451
C SEC EXAMPLE PAGE 64	C 7452
NRET = NX - NCN	C 7453
CALL ZERR (T,NRET,NRET,KT)	C 7454
CALL KLEADD (1,00,C,IVEC,JVEC,T,NCN,NX,NRET,NRET,KC,KT)	C 7455
C	C 7456
DO 50 I=1,NX	C 7457
IF (JVEC(I) .EQ. 0) GO TO 50	C 7458
NR = JVEC(I)	C 7459
T(I,NR) = 1.00	C 7460
50 CONTINUE	C 7461
C	C 7462
IF (.NOT.LEOU) GO TO 140	C 7463
WRITE(NBT,1005)	C 7464
1005 FORMAT (' SIMILARITY TRANSFORMATION MATRIX IS ')	C 7465
CALL WRITES(T,NRET,NRET,KT)	C 7466
140 CONTINUE	C 7467
C	C 7468
NAXX = NXSS + NBTG	C 7469
NXY = NEX + NAXX	C 7470
C	C 7471
THE FOLLOWING SECTION IDENTIFIES Y* VARIABLES.	C 7472
C	C 7473
K = 0	C 7474
DO 60 I=1,NJV	C 7475
IF (INDEP(I) .EQ. 0) GO TO 60	C 7476
K=K+1	C 7477
IV(K) = I	C 7478
60 CONTINUE	C 7479
C	C 7480
K=0	C 7481
DO 70 I=1,NX	C 7482
IF (JVEC(I) .EQ. 0) GO TO 70	C 7483
K=K+1	C 7484
JV(K) = IV(I)	C 7485
70 CONTINUE	C 7486
C	C 7487
NZ = NY2 + ND2 + NXSS + NBTG	C 7488
C	C 7489
DO 90 I=1,NZ	C 7490
90 IV(I) = JV(I)	C 7491
C	C 7492
DO 100 I=1,NXSS	C 7493
L=NY2+I	C 7494
K=L+ND2	C 7495
100 IV(L) = JV(K)	C 7496
C	C 7497
IF (ND2.EQ.0) GO TO 111	C 7498
DO 110 I=1,ND2	C 7499

	K = NY2 + 1	0 7500
	L = K + NXSS	0 7501
110	IV(L) = JV(K)	0 7502
111	CONTINUE	0 7503
C		0 7504
	CALL PAGEMD	C 7505
	LELO = 2*NB + 2	C 7506
	NECP = NEO + NAUX	0 7507
	WRITE (NCT,2001)	C 7508
	CALL WRITIS (INDEP,1,NECP,1)	0 7509
	WRITE (NCT,2010)	0 7510
	CALL WRITIS (LENU ,1,LELO,1)	C 7511
	WRITE (NCT,2011)	0 7512
	CALL WRITIS (LOCU ,1,LELO,1)	C 7513
	WRITE (NCT,1002)	0 7514
	CALL WRITIS (IV,1,NZ,1)	0 7515
C		0 7516
C		0 7517
	RETURN	0 7518
	END	0 7519
	SUBROUTINE Tftype (A,Z,B,NA,NZ,ITYPE,JCOL,NBKP,<BKP>,KA,KZ)	0 7520
C		0 7521
C		0 7522
	IMPLICIT REAL*8(A-H,O-Z)	0 7523
C		0 7524
	COMMON /LDSIZE/	C 7525
2	NX, NY, NDLTA, NXSS, NB, NJQ, NY2, NDZ	0 7526
	LOGICAL DEBUG,LEQU	0 7527
	COMMON /LDEBUG/ DEBUG(120)	0 7528
	EQUIVALENCE (LEQU,DEBUG(67))	0 7529
	DATA NOT/6/	0 7530
C		0 7531
	DIMENSION A(KA,1), Z(KZ,1), B(1), KOKP(1)	0 7532
C		0 7533
C	----- SUBROUTINE ARGUMENT DESCRIPTIONS -----	0 7534
C		0 7535
C	A = INPUT PARTIAL DERIVATIVE MATRIX	0 7536
C	Z = OUTPUT REDUCED PARTIAL DERIVATIVE MATRIX. (NZ,NZ)	C 7537
C	B = OUTPUT VECTOR OF COEFF. FOR DESIRED TF INPUT. (NZ,1)	C 7538
C	NA = INPUT SIZE OF A.	0 7539
C	NZ = OUTPUT SIZE OF Z	0 7540
C	ITYPE = INPUT	C 7541
C	1 FORWARD PATH TF XSS(I)/RT(J)	0 7542
C	2 FEEDBACK TF B(I)/XSS(J)	0 7543
C	3 OPEN LOOP TF B(I)/RT(J)	0 7544
C	4 OPEN LOOP TF XSS(I)/RS(J)	0 7545
C	5 CLOSED LOOP TF XSS(I)/RT(J)	0 7546
C	6 CLOSED LOOP TF XSS(I)/RS(J)	0 7547
C	7 PARTIAL OPEN LOOP B(I)/RI(J)	0 7548
C	8 SPECIAL CASE SEE BELOW	0 7549
C		0 7550
C	NOTE-- A MINUS SIGN ON ITYPE INDICATES	0 7551
C	NEGATIVE FEEDBACK FOR NUMERATOR	0 7552
C	AUGMENTATION SELECTION OF PROPER B.	0 7553
C		

C	JCOL = INPUT COL LOCATION IN A OF DESIRED INPUT(J). LOCAL	0 7554
C	FOR SENSOR SIGNALS	0 7555
C	0 < JCOL < NXSS+1	0 7556
C	FOR TORQUE SIGNALS	0 7557
C	0 < JCOL < NB+1	0 7558
C	NBKP = INPUT NO. OF B'S TO RETAIN ITYPE=7	0 7559
C	KEKP = INPUT ID VECTOR NOTING WHICH B'S TO KEEP (LOCAL)	0 7560
C	KA = INPUT ROW DIMENSION OF A IN CALLING PROGRAM	0 7561
C	KZ = INPUT ROW DIMENSION OF Z IN CALLING PROGRAM	0 7562
C		0 7563
C	ESTABLISH LEADING ELE LOCATORS FOR EACH PARTITION OF A	0 7564
C	ASSUMED ORDER IS Y,XSS,DELTA,B	0 7565
C		0 7566
C	LY = LEADING ELEMENT FOR STATE VARIABLES	0 7567
C	LX = LEADING ELEMENT FOR SENSOR SIGNALS	0 7568
C	LC = LEADING ELEMENT FOR CONTROL STATE VARIABLES	0 7569
C	LE = LEADING ELEMENT FOR TORQUE SIGNALS	0 7570
C	XSN = +1,-1, POSITIVE, NEGATIVE FEEDBACK	0 7571
C		0 7572
	LY = 1	0 7573
	LX = LY + NY2	0 7574
	LC = LX + NXSS	0 7575
	LB = LC + ND2	0 7576
	XSN = ISIGN(1,ITYPE)	0 7577
	ITYPE = ABS(ITYPE)	0 7578
C		0 7579
	NERROR = 1	0 7580
	IF (ITYPE .LT. 1 .OR. ITYPE .GT. 8) GO TO 999	0 7581
	NERROR = 2	0 7582
	IF (JCOL .LT. 0 .OR. JCOL .GT. NA) GO TO 999	0 7583
C		0 7584
	GO TO (1,2,3,4,5,6,7,9), ITYPE	0 7585
C		0 7586
	1 CONTINUE	0 7587
C	-----ITYPE = 1 -----	0 7588
C	TYPE 1 FORWARD PATH TRANSFER FUNCTION FOR PLANT WITH NO	0 7589
C	FEEDBACK, EQUATION III-49, VOL 1	0 7590
C		0 7591
	FORM Z = A11,A12	0 7592
C	A21,A22	0 7593
C		0 7594
	B = A14(JCOL)*XSN	0 7595
C	A24(JCOL)*XSN	0 7596
C		0 7597
	NZ = NY2 + NXSS	0 7598
C	KCOL = ABSOLUTE LOCATION OF TORQUE SIGNAL	0 7599
	KCOL = LE-1+JCOL	0 7600
	DO 10 I=1,NZ	0 7601
	E(I) = A(I,KCOL) * XSN	0 7602
	DO 10 J=1,NZ	0 7603
	10 Z(I,J) = A(I,J)	0 7604
	IF (.NOT.LEQU) RETURN	0 7605
	WRITE (NCT,1000) ITYPE	0 7606
	WRITE (NCT,1001)	0 7607
	CALL WRITES(Z,NZ,NZ,KZ)	0 7608
	WRITE (NCT,1002)	0 7609
	CALL WRITES(B,1,NZ,1)	0 7610
	RETURN	0 7611
C		0 7612
	2 CONTINUE	0 7613

C	-----ITYPE = 2 -----	0 7614
C	TYPE 2 FEEDBACK PATH H(S) FOR THE CONTROLLER ONLY. THE TRANSFER	0 7615
C	FUNCTION RELATES CONTROL SYSTEM OUTPUT TO SENSOR SIGNAL	0 7616
C	INPUT EQUATION III-53, VOL 1	0 7617
C		0 7618
C	FORM Z = A33,A34	0 7619
C	A43,A44	0 7620
C		0 7621
C	B = A32(JCOL)	0 7622
C	A42(JCOL)	0 7623
C		0 7624
C	NZ = ND2 + NB	0 7625
C	KCOL = ABSOLUTE LOCATION OF SENSOR SIGNAL	0 7626
C	KCOL = LX-1+JCOL	0 7627
C	CALL ZERG (Z,NZ,NZ,KZ)	0 7628
C	DO 20 I=1,NZ	0 7629
C	IRA = I + NY2 + NXSS	0 7630
C	B(I) = A(IRA,KCOL)	0 7631
C	DO 20 J=1,NZ	0 7632
C	JCA = J + NY2 + NXSS	0 7633
C	Z(I,J) = A(IRA,JCA)	0 7634
C	20 CONTINUE	0 7635
C	IF(.NOT.LEQU) RETURN	0 7636
C	WRITE (NCT,1000) ITYPE	0 7637
C	WRITE (NCT,1001)	0 7638
C	CALL WRITES(Z,NZ,NZ,KZ)	0 7639
C	WRITE (NCT,1002)	0 7640
C	CALL WRITES(B,1,NZ,1)	0 7641
C	RETURN	0 7642
C		0 7643
C	3 CONTINUE	0 7644
C	-----ITYPE = 3 -----	0 7645
C	TYPE 3 CLASSICAL OPEN LOOP TRANSFER FUNCTION RELATES CONTROL	0 7646
C	SYSTEM OUTPUT TO EXTERNAL PLANT INPUT. EQUATION III-55	0 7647
C		0 7648
C	FORM Z = A11,A12,A13, 0	0 7649
C	A21,A22,A23, 0	0 7650
C	A31,A32,A33,A34	0 7651
C	A41,A42,A43,A44	0 7652
C		0 7653
C	B = A14(JCOL)*XSN	0 7654
C	A24(JCOL)*XSN	0 7655
C	0	0 7656
C	0	0 7657
C		0 7658
C	NZ = NY2 + NXSS + ND2 + NB	0 7659
C	KCOL = ABSOLUTE LOCATION OF TORQUE SIGNAL	0 7660
C	KCOL = LE-1+JCOL	0 7661
C	CALL ZERG (Z,NZ,NZ,KZ)	0 7662
C	M = NY2 + NXSS	0 7663
C	DO 25 I=1,NZ	0 7664
C	E(I) = A(I,KCOL)*XSN	0 7665
C	DO 25 J=1,NZ	0 7666
C	25 Z(I,J) = A(I,J)	0 7667
C	DO 26 I=1,M	0 7668
C	DO 26 J=LB,NZ	0 7669
C	26 Z(I,J) = 0.00	0 7670
C	DO 30 I=LB,NZ	0 7671
C	30 E(I) = 0.00	0 7672
C	IF(.NOT.LEQU) RETURN	0 7673

```

WRITE (NCT,1000) ITYPE                                0 7674
WRITE (NCT,1001)                                        0 7675
CALL WRITES(Z,NZ,NZ,KZ)                                0 7676
WRITE (NCT,1002)                                        0 7677
CALL WRITES(B,1,NZ,1)                                  0 7678
RETURN                                                  C 7679

C                                                       0 7680
4 CONTINUE                                             0 7681
C      -----ITYPE = 4 -----                        0 7682
C      TYPE 4 OPEN LOOP TRANSFER FUNCTION, RELATES PLANT SENSOR SIGNAL 0 7683
C      OUTPUT TO CONTROLLER NOISE INPUT, EQUATION 111-57 0 7684
C                                                       0 7685
C      FORM Z = A11,A12,A13,A14                        0 7686
C               A21,A22,A23,A24                        0 7687
C               A31, 0,A33,A34                          0 7688
C               A41, 0,A43,A44                          0 7689
C                                                       0 7690
C      B = 0                                             0 7691
C               0                                         0 7692
C               A32(JCOL)                                0 7693
C               A42(JCOL)                                0 7694
C                                                       0 7695
C      NZ = NY2 + NXSS + NUZ + NB                       0 7696
C      KCOL = ABSOLUTE LOCATION OF SENSOR SIGNAL        0 7697
C      KCOL = LX-1+JCOL                                  0 7698
C      CALL Z-FC (Z,NZ,NZ,KZ)                           0 7699
C      M = NY2 + NXSS                                    0 7700
C      DO 35 I=1,NZ                                       0 7701
C      E(I) = A(I,KCOL)                                   0 7702
C      DO 35 J=1,NZ                                       0 7703
35  Z(I,J) = A(I,J)                                       0 7704
C      DO 36 I=1,D,NZ                                     0 7705
C      DO 36 J=LX,M                                       0 7706
36  Z(I,J) = 0.00                                         0 7707
C      DO 40 I=1,M                                       0 7708
40  E(I) = 0.00                                           0 7709
C      IF (.NOT.LEOU) RETURN                             0 7710
C      WRITE (NCT,1000) ITYPE                            0 7711
C      WRITE (NCT,1001)                                  0 7712
C      CALL WRITES(Z,NZ,NZ,KZ)                           0 7713
C      WRITE (NCT,1002)                                  0 7714
C      CALL WRITES(B,1,NZ,1)                              0 7715
C      RETURN                                             0 7716
C                                                       0 7717
5 CONTINUE                                             0 7718
C      -----ITYPE = 5 -----                        0 7719
C      TYPE 5 TRANSFER FUNCTION THAT RELATES SENSOR SIGNAL OUTPUT    0 7720
C      TO EXTERNALLY APPLIED PLANT INPUTS WITH THE CONTROL          0 7721
C      SYSTEM ENTIRELY ACTIVE, EQUATION 111-59 0 7722
C                                                       0 7723
C      FORM Z = A11,A12,A13,A14                        0 7724
C               A21,A22,A23,A24                        0 7725
C               A31,A32,A33,A34                        0 7726
C               A41,A42,A43,A44                        0 7727
C                                                       0 7728
C      J = A14(JCOL)*XSN                                   0 7729
C      A24(JCOL)*XSN                                     0 7730
C      0                                                  0 7731
C      0                                                  0 7732
C      NZ = NY2 + NXSS + NUZ + NB                       0 7733

```


C	KCOL = ABSOLUTE LOCATION OF TORQUE SIGNAL	0 7734
	KCOL = LB-1+JCOL	0 7735
	CALL ZERC (Z,NZ,NZ,KZ)	0 7736
	DO 45 I=1,NZ	0 7737
	E(I) = A(I,KCOL)*XSN	0 7738
	DO 45 J=1,NZ	0 7739
45	Z(I,J) = A(I,J)	0 7740
	DO 50 I=LD,NZ	0 7741
50	B(I) = 0.00	0 7742
	IF(.NOT.LEQU) RETURN	0 7743
	WRITE (NCT,1000) ITYPE	0 7744
	WRITE (NCT,1001)	0 7745
	CALL WRITES(Z,NZ,NZ,KZ)	0 7746
	WRITE (NCT,1002)	0 7747
	CALL WRITES(B,1,NZ,1)	0 7748
	RETURN	0 7749
C		0 7750
6	CONTINUE	0 7751
C	-----ITYPE = 6 -----	0 7752
C	TYPE 6 CLOSED LOOP TRANSFER FUNCTION RELATES PLANT SENSOR	0 7753
C	SIGNAL OUTPUT TO SENSOR SIGNAL NOISE INPUT ALL CONTROL	0 7754
C	SYSTEM LOOPS ACTIVE, EQUATION III-61	0 7755
C		0 7756
C	FCR# Z = A11,A12,A13,A14	0 7757
C	A21,A22,A23,A24	0 7758
C	A31,A32,A33,A34	0 7759
C	A41,A42,A43,A44	0 7760
C		0 7761
C	B = 0	0 7762
C	0	0 7763
C	A32(JCOL)	0 7764
C	A42(JCOL)	0 7765
C		0 7766
	NZ = NY2 + NXSS + ND2 + NB	0 7767
C	KCOL = ABSOLUTE LOCATION OF SENSOR SIGNAL	0 7768
	KCOL = LX-1+JCOL	0 7769
	CALL ZERC (Z,NZ,NZ,KZ)	0 7770
	M = NY2 + NXSS	0 7771
	DO 55 I=1,NZ	0 7772
	E(I) = A(I,KCOL)	0 7773
	DO 55 J=1,NZ	0 7774
55	Z(I,J) = A(I,J)	0 7775
	DO 60 I=1,M	0 7776
60	E(I) = 0.00	0 7777
	IF(.NOT.LEQU) RETURN	0 7778
	WRITE (NCT,1000) ITYPE	0 7779
	WRITE (NCT,1001)	0 7780
	CALL WRITES(Z,NZ,NZ,KZ)	0 7781
	WRITE (NCT,1002)	0 7782
	CALL WRITES(B,1,NZ,1)	0 7783
	RETURN	0 7784
C		0 7785
C		0 7786
7	CONTINUE	0 7787
C	-----ITYPE = 7-----	0 7788
C	TYPE 7 QUASI OPEN LOOP TRANSFER FUNCTION RELATES CONTROL SYSTEM	0 7789
C	OUTPUTS DUE TO PLANT VARIABLE INPUTS WITH THE OPTION TO	0 7790
C	OPEN SOME CONTROLLER OUTPUT CHANNELS. MAX NUMBER OF	0 7791
C	FEEDBACK SIGNALS IS 3, SEE PAGE 77,VOL I	0 7792
C	NBKP = NUMBER OF FEEDBACK SIGNALS	0 7793

```

C          KUKP(1) = LOCAL LOCATION OF SIGNAL TO FEEDBACK      0 7794
C                                                                0 7795
C          FCR* Z = A11,A12,A13,(A14)                          0 7796
C                   A21,A22,A23,(A24)                          0 7797
C                   A31,A32,A33,A34                             0 7798
C                   A41,A42,A43,A44                             0 7799
C                                                                0 7800
C                   B = A14(JCOL)*XSN                           0 7801
C                   A24(JCOL)*XSN                               0 7802
C                   0                                            0 7803
C                   0                                            0 7804
C                                                                0 7805
C          NZ = NY2 + NXSS + ND2 + NB                             0 7806
C          KCCL = ABSOLUTE LOCATION OF TORQUE SIGNAL            0 7807
C          KCCL = LB-1+JCOL                                       0 7808
C          CALL ZERC (Z,NZ,NZ,KZ)                                0 7809
C          M = NY2 + NXSS                                         0 7810
C          M1 = M + ND2                                           0 7811
C          DO 65 I=1,M                                           0 7812
C             E(I) = A(I,KCCL) * XSN                              0 7813
C          DO 62 J=1,M1                                           0 7814
C 62      Z(I,J) = A(I,J)                                         0 7815
C          DO 63 J=1,NBKP                                         0 7816
C          LCCL = ABSOLUTE LOCATION OF TORQUE SIGNAL TO FEEDBACK 0 7817
C          0 < KUKP(J) < NB+1                                     0 7818
C          LCCL = LB-1+KUKP(J)                                    0 7819
C 63      Z(I,LCCL) = A(I,LCCL)                                   0 7820
C 65 CONTINUE                                                    0 7821
C          DO 70 I=LD,NZ                                          0 7822
C             E(I) = 0.00                                         0 7823
C          DO 70 J=1,NZ                                          0 7824
C             Z(I,J) = A(I,J)                                     0 7825
C 70 CONTINUE                                                    0 7826
C          IF(.NOT.LEQD) RETURN                                    0 7827
C          WRITE (NCT,1000) ITYPE                                0 7828
C          WRITE (NCT,1001)                                       0 7829
C          CALL WRITES(Z,NZ,NZ,KZ)                                0 7830
C          WRITE (NCT,1002)                                       0 7831
C          CALL WRITES(B,1,NZ,1)                                  0 7832
C          RETURN                                                  0 7833
C                                                                0 7834
C                                                                0 7835
C          6 CONTINUE                                             0 7836
C          -----IABS(ITYPE) = 8 -----                          0 7837
C          TYPE 8 CLOSED LOOP TRANSFER FUNCTION BETWEEN ANY SYSTEM STATE 0 7838
C                   VARIABLE (JCOL NOT LOCAL) OUTPUT TO ANY SYSTEM STATE 0 7839
C                   VARIABLE NOISE INPUT                          0 7840
C                   ITYPE = 8 FEEDBACK LOOP CUT                  0 7841
C                   ITYPE = -8 CLOSED LOOP                        0 7842
C                                                                0 7843
C          FORM FOR ITYPE = -8                                     0 7844
C                                                                0 7845
C                   A11,A12,....,A1J,....,A1N                   -A1J 0 7846
C                   A21,A22,....,A2J,....,A2N                   -A2J 0 7847
C                   .      .      .      .                      .    0 7848
C                   .      .      .      .                      .    0 7849
C                   .      .      .      .                      .    0 7850
C          Z =      AJ1,AJ2,....,AJJ,....,AJN                   B = 0 0 7851
C                   .      .      .      .                      .    0 7852
C                   .      .      .      .                      .    0 7853

```

```

C          . . . . .
C          AN1,AN2,....,ANJ,....,ANN          -ANJ
C
C      WHERE:
C          J = JCOL
C          N = NZ
C          ELEMENTS OF Z EQUATED TO A MATRIX
C          ELEMENTS OF B EQUATED TO NEGATIVE OF COLUMN JCOL
C          OF A WITH ROW JCOL OF B SET EQUAL TO ZERO
C
C      FORM FOR ITYPE = 8
C
C          A11,A12,...., 0 ,....,A1N          -A1J
C          A21,A22,...., 0 ,....,A2N          -A2J
C          . . . . .
C          . . . . .
C          . . . . .
C      Z =  AJ1,AJ2,....,AJJ,....,AJN          B =  0
C          . . . . .
C          . . . . .
C          . . . . .
C          AN1,AN2,...., 0 ,....,ANN          -ANJ
C
C      WHERE:
C          J = JCOL
C          N = NZ
C          ELEMENTS OF Z EQUATED TO A MATRIX, COLUMN JCOL RESET
C          TO ZERO, (JCOL,JCOL) ELEMENT OF Z
C          RESET AGAIN TO A(JCOL,JCOL)
C          ELEMENTS OF B SAME AS IN ITYPE = -3
C
CCC  FOR ITYPE = 8, LRY(2,ICYC) IS GLOBAL REFERENCE INPUT STATE OF Y*,
CCC          ((WHOSE FEEDBACK LOOP IS CUT)).
CCC          LRY(3,ICYC) IS GLOBAL OUTPUT STATE OF Y*
C
CCC  FOR ITYPE = -8, LRY(2,ICYC) AND LRY(3,ICYC) ARE THE SAME
CCC          (AS ITYPE = 8), BUT THIS IS A COMPLETELY CLOSED
CCC          LOOP TRANSFER FUNCTION.
C
      NZ = NY2 + NXSS + ND2 + NB
      DO 75 I=1,NZ
      E(I) = -A(I,JCOL)
      DO 75 J=1,NZ
75  Z(I,J) = A(I,J)
      IF (XSN .LT. 0.00) GO TO 77
      DO 76 I=1,NZ
76  Z(I,JCOL) = 0.00
77  Z(JCOL,JCOL) = -B(JCOL)
      E(JCOL) = 0.00
      IF (.NOT.LEQU) RETURN
      WRITE (NCT,1000) ITYPE
      WRITE (NCT,1001)
      CALL WRITES(Z,NZ,NZ,KZ)
      WRITE (NCT,1002)
      CALL WRITES(B,1,NZ,1)
      RETURN
C
C
999  CONTINUE
      WRITE (6,2001) NERRCK

```

```

C 1000 FORMAT (' SUBROUTINE TYPE, TRANSFER FUNCTION TYPE',I3) 0 7914
C 1001 FORMAT (' THE CHARACTERISTIC MATRIX IS ') 0 7915
C 1002 FORMAT (' THE CORRESPONDING INPUT COEFFICIENTS ARE ') 0 7916
C 1003 FORMAT (I11,3X,48HPROGRAM STOPPED IN SUBROUTINE TYPE.  NERRR = 0 7918
* , I3) 0 7919
SICP 0 7920
END 0 7921

SUBROUTINE NUMS (A,D,B,K1R,R1I,R2R,P2I,PTOL, 0 7922
* GAIN,IFLG,NN,NZRC,IY,NA,KA) 0 7923
0 7924
0 7925
0 7926
0 7927
0 7928
0 7929
0 7930
0 7931
0 7932
0 7933
0 7934
0 7935
0 7936
0 7937
0 7938
0 7939
0 7940
0 7941
0 7942
0 7943
0 7944
0 7945
0 7946
0 7947
0 7948
0 7949
0 7950
0 7951
0 7952
0 7953
0 7954
0 7955
0 7956
0 7957
0 7958
0 7959
0 7960
0 7961
0 7962
0 7963
0 7964
0 7965
0 7966
0 7967
0 7968
0 7969
0 7970
0 7971
0 7972
0 7973
0 7974
0 7975
0 7976
0 7977
0 7978
0 7979
0 7980
0 7981
0 7982
0 7983
0 7984
0 7985
0 7986
0 7987
0 7988
0 7989
0 7990
0 7991
0 7992
0 7993
0 7994
0 7995
0 7996
0 7997
0 7998
0 7999

```

C	FORM AUGMENTED A MATRIX	0 7968
C	REPLACE COL IY OF A WITH COL NA OF A AND	0 7969
C	PUT E INTO COL NA OF A	0 7970
C		0 7971
	DO 10 I=1,NA	0 7972
	A(I,IY) = A(I,NA)	0 7973
10	A(I,NA) = -B(I)	0 7974
C	INTERCHANGE ROW IY OF A WITH ROW NA OF A	0 7975
	DO 15 J=1,NA	0 7976
	Z = A(IY,J)	0 7977
	A(IY,J) = A(NA,J)	0 7978
15	A(NA,J) = Z	0 7979
C	OBTAIN PIVOTAL ELE AND CK FOR ZERO	0 7980
	Z = A(NA,NA)	0 7981
	IF (Z .EQ. 0.00) GO TO 50	0 7982
	GAIN = -Z	0 7983
	NN = 1	0 7984
	DO 20 J=1,NA	0 7985
20	A(NA,J) = A(NA,J) / Z	0 7986
	NA2 = NA-1	0 7987
	DO 30 I=1,NA2	0 7988
	DO 30 J=1,NA	0 7989
30	A(I,J) = A(I,J) - A(I,NA) * A(NA,J)	0 7990
	CALL WRITE (A,NA,NA,4HNUM,KA)	0 7991
C	GET ROOTS OF AUGMENTED A MATRIX IF ALL PIVOT ELEMENTS NON-ZERO	0 7992
	CALL QRDQVR (A,NA2,R1R,R1I,KA)	0 7993
	NZRO = NA2	0 7994
	GO TO 100	0 7995
C	COME HERE IF PIVOT ELEMENT ZERO,SEE BOTTOM PAGE 71,VOL 1	0 7996
50	CONTINUE	0 7997
C		0 7998
	NN = 1	0 7999
	NA2 = NA-1	0 8000
C		0 8001
C	FORM (A - I*CON)	0 8002
C		0 8003
	DO 55 I=1,NA2	0 8004
55	A(I,I) = A(I,I) - CON	0 8005
C		0 8006
C	CALL GAUSSI TO GET (A - I*CON)**(-1) RETURN IT IN D ARRAY	0 8007
	CALL GAUSSI (A,D,NA,KA)	0 8008
	IF (IFL2 .EQ. 0) GO TO 57	0 8009
	IFL2 = 0	0 8010
	IFLG = 0	0 8011
C		0 8012
C	ZERO GAIN ENCOUNTERED IN GAUSSI.	0 8013
C		0 8014
	RETURN	0 8015
C		0 8016
57	CONTINUE	0 8017
C		0 8018
C	FORM PRODUCT DEFINED BY EQUATION III-41	0 8019
C	NOTE THE I TILDA MATRIX HAS ZERO IN NA,NA POSITION	0 8020
	DO 59 I=1,NA	0 8021
59	C(I,NA) = 0.00	0 8022
C		0 8023
C	FIND ROOTS OF EQUATION III-42	0 8024
	IF(LEQU) CALL WRITE(D,NA,NA,6H -D- ,KA)	0 8025
	CALL QRDQVR (D,NA,R2R,R2I,KA)	0 8026
C		0 8027

C	REMOVE THE $P(1) = 0$. ROOTS.	0 8028
C		0 8029
	CALL SIFT (R2R,NA,PTOL)	0 8030
	CALL SIFT (R2I,NA,PTOL)	0 8031
C		0 8032
	K=C	0 8033
	DO 60 I=1,NA	0 8034
	IF (R2R(I) .EQ. 0.00 .AND.	0 8035
	* R2I(I) .EQ. 0.00) GO TO 60	0 8036
	K=K+1	0 8037
	R1R(K) = R2R(I)	0 8038
	R1I(K) = R2I(I)	0 8039
	60 CONTINUE	0 8040
C		0 8041
C	GET NUMERATOR GAIN SEE EQUATION III-48	0 8042
C	NZRO = NUMBER OF NON-ZERO ROOTS IN NUMERATOR EQUATION	0 8043
	NZRO = K	0 8044
	DO 70 I=1,NA	0 8045
	IF (I .GT. NZRO) GO TO 71	0 8046
	IF (R1I(I) .EQ. 0.00) GO TO 65	0 8047
	DRVEC(I) = DRVEC(I)*DSQRT(R1R(I)**2 + R1I(I)**2)	0 8048
	GO TO 70	0 8049
	65 DRVEC(I) = DRVEC(I)*R1R(I)	0 8050
	70 CONTINUE	0 8051
C	FORM PRODUCT DEFINED BY EQUATION III-48 TO OBTAIN	0 8052
C	NUMERATOR GAIN	0 8053
	71 GAIN = DRVEC(1)	0 8054
	DO 58 I=2,NA	0 8055
	58 GAIN = GAIN*DRVEC(I)	0 8056
	IF (MOD((NA-NZRO),2) .EQ. 1) GAIN = -GAIN	0 8057
C		0 8058
C	REMOVE THE SHIFT VALUE TO OBTAIN TRUE ROOTS.	0 8059
C	S(1) = 1.0/P(1) + CON S AND P COMPLEX OF COURSE	0 8060
C		0 8061
	DO 80 I=1,NZRO	0 8062
	RMOD = R1R(I)**2 + R1I(I)**2	0 8063
	R1R(I) = R1R(I)/RMOD + CON	0 8064
	80 R1I(I) = -R1I(I)/RMOD	0 8065
C		0 8066
	CALL WRITE (DRVEC,1,NA,QUADRATIC,1)	0 8067
100	CONTINUE	0 8068
	IF (LEDD) CALL WRITE(R1R,1,NZRO,6H -R1R-,1)	0 8069
	IF (LEGU) CALL WRITE(R1I,1,NZRO,6H -R1I-,1)	0 8070
	IF (LEGU) CALL WRITE(GAIN,1,1,6H GAIN ,1)	0 8071
	RETURN	0 8072
	END	0 8073
	EXECUTING TPLY (A,B,C,X,NS,L)	0 8074
C	DEBUG # 69	0 8075
	IMPLICIT REAL*8 (A-H,O-Z)	0 8076
C		0 8077
	DIMENSION A(1), B(1)	0 8078
C		0 8079
C	SUBROUTINE CONVERTS TRANSFER FUNCTION POLYNOMIAL	0 8080
C	EXPRESSIONS TO FIRST ORDER CANONICAL STATE	0 8081

```

C      SPACE FORM AND RETURNS THE TRANSFORMED OUTPUT          0 8082
C      VARIABLE, X.                                           0 8083
C                                                             0 8084
C                                                             0 8085
C      U -----> [ TF(S) ] -----> X                    0 8086
C                                                             0 8087
C                                                             0 8088
C                                                             0 8089
C                                                             0 8090
C                                                             0 8091
C                                                             0 8092
C                                                             0 8093
C      B(1) + B(2)*S + B(3)*S**2 + ... + B(KS)*S**(KS-1)    C 8094
C      TF(S) = -----                                         0 8095
C      A(1) + A(2)*S + A(3)*S**2 + ... + A(NS)*S**(NS-1)    0 8096
C                                                             0 8097
C      WHERE KS.LE.NS                                         0 8098
C                                                             0 8099
C      -----SUBROUTINE ARGUMENT DESCRIPTIONS-----        0 8100
C                                                             0 8101
C      A      = INPUT VECTOR OF DENOMINATOR POLYNOMIAL        0 8103
C                COEFFICIENTS--ASCENDING ORDER.              0 8104
C      B      = INPUT VECTOR OF NUMERATOR POLYNOMIAL          0 8105
C                COEFFICIENTS--ASCENDING ORDER.              0 8106
C      U      = INPUT STATE VARIABLE TO BE OPERATED ON BY THE 0 8107
C                POLYNOMIAL TRANSFER FUNCTION.                0 8108
C      X      = OUTPUT VARIABLE RESULTING FROM THE TRANSFER    0 8109
C                FUNCTION OPERATING ON U.                      0 8110
C      NS     = INPUT SIZE OF A AND B.                         0 8111
C      L      = INPUT LOCATION (IN STATE VECTOR) OF THE       0 8112
C                LEADING ELEMENT OF THE NS-1 STATE VARIABLES  0 8113
C                ESTABLISHED FROM THE POLYNOMIALS.            0 8114
C                                                             0 8115
C      FOR THEORY SEE VOL 1,PAGE 78                            C 8116
C                                                             0 8117
C      COMMON /VECTOR/                                         0 8118
C      E      Y      (250), YD      (250)                      430 8119
C      DATA N1/ 6 /                                           0 8120
C                                                             0 8121
C      NORMALIZE A AND B COEFFICIENTS TO COEFFICIENT OF      C 8122
C      HIGHEST DERIVATIVE IN DENOMINATOR, A(NS).             0 8123
C                                                             0 8124
C      AN = A(NS)                                              0 8125
C      IF (AN .EQ. 0.00) GO TO 999                             0 8126
C                                                             0 8127
C      DO 10 I=1,NS                                           C 8128
C      A(I) = A(I) / AN                                         0 8129
C      10 E(I) = B(I) / AN                                       0 8130
C                                                             0 8131
C      BN = E(NS)                                              0 8132
C                                                             0 8133
C      FORM STATE VECTOR TIME DERIVATIVES AND PUT INTO YDOT  0 8134
C      BEGINNING WITH LOCATION L IN YDOT.                     0 8135
C                                                             0 8136
C      CARRY OUT MATRIX OPERATIONS CALLED FOR BY EQUATION 11-69 0 8137
C                                                             0 8138
C      DO 20 I=2,NS                                           0 8139
C                                                             0 8140
C      J = NS-I+1                                             0 8141

```

```

      K = L+1-2
      IF (1.EQ. NS) GO TO 25
C
      20 YD(K) = -A(J)*Y(L) + Y(K+1) + (B(J)-A(J)*BN)*J
      25 YD(K) = -A(J)*Y(L) + (B(J)-A(J)*BN)*U
C
      X = Y(L) + BN*U
C
      RETURN
C
      999 CALL PAGEHD
      WRITE (NIT,1001)
      1001 FORMAT (///,10X,3HCOEFFICIENT OF HIGHEST ,
      *           /,10X,3HDERIVATIVE OF DENOMINATOR CANNOT ,
      *           /,10X,17HBE EQUAL TO ZERO. ,
      *           //,10X,10HPROGRAM STOPPED.)
C
      STOP
      END

```

```

0 8142
C 8143
0 8144
0 8145
0 8146
0 8147
C 8148
C 8149
0 8150
0 8151
0 8152
0 8153
0 8154
0 8155
0 8156
0 8157
C 8158
0 8159
C 8160

```

```

      SUBROUTINE UCORRT (RR,RI,N,KR,KC,KZ,RLRT,CMPR)
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      SUBROUTINE RECEIVES QR ROOT OUTPUT OF THE FORM,
C
      KR(1),RI(1),I=1,N
C
      AND PLACES REAL ROOTS (INCLUDING ZEROS) INTO
C
      MATRIX RLRT (SIZE KR), THEN PLACES THE
C
      COMPLEX PAIRS (A(I) + J E(I)) INTO
C
      MATRIX CMPR-- COMPLEX PAIR ORDER IS
C
      CMPR(I) = A(I)
C
      CMPR(I+1) = B(I)
C
      SAVES ONLY REAL AND POSITIVE IMAG PARTS IN CMPR.
C
      SIZE OF CMPR IS 2 * KC.
C
      -----SUBROUTINE ARGUMENT DESCRIPTIONS-----
C
      RR      = INPUT ARRAY OF REAL PARTS. SIZE IS N
C
      RI      = INPUT ARRAY OF IMAGINARY PARTS. SIZE IS N.
C
      N       = INPUT SIZE OF RR AND RI. NUMBER OF QR ROOTS.
C
      KR      = OUTPUT SIZE OF RLRT. NUMBER OF REAL ROOTS,
C
      (INCLUDING ZEROS).
C
      KC      = NUMBER OF COMPLEX PAIRS.
C
      KZ      = OUTPUT NUMBER OF ZEROS.
C
      RLRT    = OUTPUT REAL ROOT ARRAY. SIZE KR.
C
      CMPR    = OUTPUT COMPLEX PAIR ARRAY. SIZE 2*KC.
C
      DIMENSION RR(1),RI(1),RLRT(1),CMPR(1)
C
      KR = 0
C
      KC = 0
C
      KZ = 0
C
      DO 10 I=1,N
C
      IF (RI(I) .EQ. 0.00) GO TO 5
C
      IF (RI(I) .LT. 0.00) GO TO 10

```

```

0 8161
0 8162
0 8163
0 8164
0 8165
0 8166
0 8167
0 8168
0 8169
0 8170
0 8171
0 8172
0 8173
0 8174
0 8175
0 8176
0 8177
0 8178
0 8179
0 8180
0 8181
0 8182
0 8183
0 8184
0 8185
0 8186
0 8187
0 8188
0 8189
0 8190
0 8191
0 8192
0 8193
0 8194
0 8195

```


KC = KC+1	0 8196
L = 2*KC	0 8197
CMPR(L-1) = RR(1)	0 8198
CMPR(L) = RI(1)	0 8199
GO TO 10	0 8200
5 CONTINUE	0 8201
KR = KR+1	0 8202
FLRT(KR) = RR(1)	0 8203
IF (RP(1) .EQ. 0.00) KZ = KZ+1	0 8204
10 CONTINUE	0 8205
C	0 8206
RETURN	0 8207
END	0 8208

SUBROUTINE RMVZRO (RR,NR)	0 8209
C	0 8210
IMPLICIT REAL*8(A-H,O-Z)	0 8211
C	0 8212
C SUBROUTINE REMOVES REAL ZEROS FROM REAL ROOT ARRAY.	0 8213
C	0 8214
C -----SUBROUTINE ARGUMENT DESCRIPTIONS-----	0 8215
C	0 8216
C RR = INPUT/OUTPUT ARRAY CONTAINING ALL REAL ROOTS.	0 8217
C NR = INPUT/OUTPUT NUMBER OF REAL ROOTS.	0 8218
C	0 8219
C DIMENSION RR(1)	0 8220
C	0 8221
K=0	0 8222
DO 10 I=1,NR	0 8223
IF (RR(I) .EQ. 0.00) GO TO 10	0 8224
K=K+1	0 8225
RR(K) = RR(I)	0 8226
10 CONTINUE	0 8227
NR = K	0 8228
RETURN	0 8229
END	0 8230

SUBROUTINE SIFT (A,N,TOL)	0 8231
C	0 8232
IMPLICIT REAL*8(A-H,O-Z)	0 8233
C	0 8234
C DIMENSION A(1)	0 8235
C	0 8236
C SUBROUTINE SEARCHES ARRAY, A, FOR SMALL VALUES OF A AND SETS	0 8237
C THESE SMALL VALUES TO 0.0.	0 8238
C	0 8239
C -----SUBROUTINE ARGUMENT DESCRIPTIONS-----	0 8240
C	0 8241
C A = INPUT, OUTPUT VECTOR ARRAY TO BE SCANNED FOR SMALL VALUES.	0 8242
C N = INPUT SIZE OF A.	0 8243
C TOL = INPUT TOLERANCE. IF (A(I) .LT. TOL) A(I) = 0.0	0 8243

C		0 8244
	DO 10 I=1,N	0 8245
	IF (CAUS(A(I)) .LT. TOL) A(I) = 0.00	0 8246
	10 CONTINUE	0 8247
C		0 8248
	RETURN	0 8249
	END	0 8250
	SUBROUTINE SFREQ2 (NNR,ICN,NDR,ICD,GAIN,	0 8251
1	FBRN,FBNC,FER ,FEDC ,	0 8252
2	FMIN,FMAX,TITLE)	0 8253
C		0 8254
	DEBUG # 73	0 8255
C		0 8256
	IMPLICIT REAL*8 (A-H,O-Z)	0 8257
	REAL*4 SAVED, SAVEP, SAVED, SAVEA	0 8258
	REAL*4 FMIN , FMAX , TITLE	0 8259
C		0 8260
	SUBROUTINE DETERMINES S-PLANE FREQUENCY RESPONSE	0 8261
	USING VARIABLE INCREMENTING TECHNIQUES.	0 8262
C		0 8263
	FREQUENCY RESPONSE SAVED IN COMMON BLOCK /PSTUFF/	0 8264
C		0 8265
	-----SUBROUTINE ARGUMENT DESCRIPTIONS-----	0 8266
C		0 8267
	NNR = INPUT NUMERATOR REAL ROOT COUNT.	0 8268
C		0 8269
	ICN = INPUT NUMERATOR COMPLEX PAIR ROOT COUNT.	0 8270
C		0 8271
	NDR = INPUT DENOMINATOR REAL ROOT COUNT.	0 8272
C		0 8273
	ICD = INPUT DENOMINATOR COMPLEX PAIR ROOT COUNT.	0 8274
C		0 8275
	GAIN = INPUT DODE GAIN.	0 8276
C		0 8277
	FBRN = INPUT NUMERATOR REAL ROOTS (INCLUDING ZEROS).	0 8278
C		0 8279
	FBNC = INPUT NUMERATOR COMPLEX PAIRS.	462 8280
C		0 8281
	FER = INPUT DENOMINATOR REAL ROOTS (INCLUDING ZEROS).	464 8282
C		0 8283
	FEDC = INPUT DENOMINATOR COMPLEX PAIRS.	0 8284
C		0 8285
	TITLE = INPUT 80 CHARACTER ALPHANUMERIC TITLE.	0 8286
C		0 8287
	COMPLEX*16 PRD0,F0EN,FNUM	426 8288
	DIMENSION F0RN (1) , F0NC (1) , FBR (1) , F0JC (1) ,	0 8289
1	TITLE(1) , WD(207) , TABG(20) , TABJP(31),	0 8290
2	TABZ(30) ,TABDN(31) , PRCD(1) ,	0 8291
3	FNUM(100), F0EN(100)	0 8292
	COMMON /PSTUFF/	0 8293
	* SAVED(500), SAVEP(500), SAVED(500), SAVEA(500), KSAVE	447 8294
	COMMON /K0SIZE/	0 8295
1	KR, KRI, KRX, KVI, KV2, KVX	0 8296
	COMMON / LV1 /	0 8297
C		0 8298
	V1 (100), V2 (100), V3 (100)	426 8288
	COMMON /L0DEBUG/ L0LUG(120)	0 8289
	LOGICAL L0BUG,L0QU	0 8290
	EQUIVALENCE (L0QU,L0BUG(73))	0 8291
C		0 8292
		0 8293
C		466 8294
	DATA KWD/207/	468 8295
	DATA K0SAVE/500/	0 8296
	DATA N0T/ 6 /	0 8297
C	TABG = CROSS CONSTANT TABLE	

C		C 8298
	DATA KTABG,KTABUP,KTABDN/18,31,29/	0 8299
C		0 8300
	DATA TABLP /	0 8301
1	0.6000000, 0.7000000, 0.7500000, 0.8000000, 0.8400000,	0 8302
2	0.8800000, 0.9000000, 0.9200000, 0.9400000, 0.9600000,	0 8303
3	0.9650000, 0.9700000, 0.9750000, 0.9800000, 0.9840000,	0 8304
4	0.9880000, 0.9900000, 0.9920000, 0.9940000, 0.9960000,	0 8305
5	0.9980000, 0.9988000, 0.9995000, 0.9997500, 0.9999000,	0 8306
6	0.9999200, 0.9999400, 0.9999600, 0.9999800, 0.9999900,	C 8307
7	1.0000000/	0 8308
C		C 8309
	DATA TABG / 1.0000, 1.1000, 1.2500, 1.4000, 1.5000, 1.8000,	0 8310
1	2.0000, 2.2000, 2.5000, 2.8000, 3.2000, 3.8000,	0 8311
2	4.5000, 5.2000, 6.2000, 7.0000, 7.8000, 8.9000,	0 8312
3	0.0000, 0.0000/	0 8313
C		0 8314
C	TABUF --- LEADING UP TO THE DAMPED NATURAL FREQUENCIES.	C 8315
C		0 8316
	VMAX = 0.00	0 8317
	FDPHI = 0.00	0 8318
	W1 = 0.00	0 8319
C		0 8320
	KWCT = 0	0 8321
	KOPH = 0	0 8322
	KPRINT = 1	0 8323
	KSAVE = 0	0 8324
C		0 8325
	NRN = NNR	0 8326
	NCN = ICN	0 8327
	NRD = NDR	0 8328
	NCD = ICD	0 8329
	FK = GAIN	0 8330
C		0 8331
C		0 8332
	DO 110 I=1,KTABDN	0 8333
	J = KTABUP-I	0 8334
110	TABDN(I) = 1.00/TABUP(J)	0 8335
C		0 8336
C	TABDN --- LEADING AWAY FROM DAMPED NATURAL FREQUENCIES	0 8337
C	LNCTR --- LINE COUNTER	0 8338
C		0 8339
120	LNCTR = 50	0 8340
	JX = 13	0 8341
C		0 8342
C	NULL WD	0 8343
C		0 8344
	DO 125 I= 1,KWD	0 8345
125	WD(I) = 0.00	0 8346
130	CONTINUE	0 8347
	DO 140 I=1,KR	0 8348
	FNUM(I) = (0.00,0.00)	0 8349
	FDEN(I) = (0.00,0.00)	0 8350
140	CONTINUE	0 8351
C		0 8352
C	FMIN = LOWER LIMIT.	0 8353
C	SAVE IT AND DESTROY SAVLO IF NEEDED	0 8354
	SAVLC = DBLE(FMIN)	0 8355
C		0 8356
C	COMPUTE NUMERATOR DAMPED NATURAL FREQUENCIES.	0 8357

C	DELETE FREQUENCIES WHICH ARE HIGHLY DAMPED	0 8358
C	IF (2*ZETA**2.GT.1.) SKIP ASSOCIATED FREQUENCY	0 8359
C		0 8360
	I = 0	0 8361
	IF (NCN) 1220, 180, 190	0 8362
150	NTCTN = NCN*2	0 8363
	DO 170 J=1,NTCTN,2	0 8364
	ABLE = FBNC(J) * FBNC(J+1)	0 8365
	BAKER = FBNC(J+1) * DSQRT(1.00 - FBNC(J)**2)	0 8366
	TEMP = (BAKER)**2 - (ABLE)**2	0 8367
	IF (TEMP) 170, 170, 100	0 8368
160	I = I+1	0 8369
	WD(I) = BAKER	0 8370
170	CONTINUE	0 8371
C		0 8372
C	COMPUTE DENOMINATOR DAMPED NATURAL FREQUENCIES.	0 8373
C		0 8374
180	IF (NCD) 1240, 220, 190	0 8375
190	NTCTD = NCD*2	0 8376
C		0 8377
	DO 210 J=1,NTCTD,2	0 8378
	ABLE = FBDC(J) * FBDC(J+1)	0 8379
	BAKER = FBDC(J+1) * DSQRT(1.00 - FBDC(J)**2)	0 8380
	TEMP = (BAKER)**2 - (ABLE)**2	0 8381
	IF (TEMP) 210, 210, 200	0 8382
200	I = I+1	0 8383
	WD(I) = BAKER	0 8384
210	CONTINUE	0 8385
220	KCCUNT = I	0 8386
C		0 8387
C	THERE ARE KCCUNT FREQUENCIES.	0 8388
C	SORT THEM IN INCREASING MAGNITUDE.	0 8389
C		0 8390
	IF (KCCUNT - 1) 240, 350, 250	0 8391
240	J=1	0 8392
	GO TO 370	0 8393
250	DO 270 J=1,KCCUNT	0 8394
	DO 270 I=J,KCCUNT	0 8395
	IF (WD(J) .LE. WD(I)) GO TO 270	0 8396
	TEMP = WD(J)	0 8397
	WD(J) = WD(I)	0 8398
	WD(I) = TEMP	0 8399
270	CONTINUE	0 8400
C		0 8401
C	SORT COMPLETE.	0 8402
C	CHECK FOR EQUAL FREQUENCIES.	0 8403
C	IF SO, ELIMINATE ONE.	0 8404
C		0 8405
280	I = 1	0 8406
	J = 2	0 8407
C		0 8408
290	IF (WD(I) - WD(J)) 300, 320, 340	0 8409
300	I = I+1	0 8410
	J = J+1	0 8411
310	IF (KCCUNT - J) 350, 350, 290	0 8412
320	DO 330 K=J,KCCUNT	0 9413
	WD(K-1) = WD(K)	0 8414
330	CONTINUE	0 8415
	WD(KCCUNT) = 0.00	0 8416
	KCCUNT = KCCUNT - 1	0 8417

GO TO 310	0 8418
340 CALL PAGEHD	0 8419
WRITE (NCT,1313)	0 8420
1313 FORMAT (//10X,23HTHE SORT ROUTINE FAILED /	0 8421
1 10X,16HPROGRAM STOPPED.)	0 8422
STLP	0 8423
350 CONTINUE	0 8424
IF (LEGO) CALL WRITE(WD,1,KCOUNT,6H WD .1)	0 8425
C	0 8426
C	0 8427
C	0 8428
360 I = 1	0 8429
J = 1	0 8430
IF (WD(1) .GT. 0.00) GO TO 430	0 8431
370 W = TABG(J) * SAVLC	0 8432
IF (W .GT. DBLE(FMAX)) GO TO 400	0 8433
IF (KTABG .GT. J) GO TO 410	0 8434
SAVLC = SAVLC * 10.00	0 8435
J = 1	0 8436
KK = 1	0 8437
GO TO 840	0 8438
C	0 8439
C THE SHOW IS OVER.	0 8440
C GETOUT.	0 8441
C	0 8442
400 KK = 6	0 8443
GO TO 1490	0 8444
410 J = J+1	0 8445
KK = 1	0 8446
GO TO 840	0 8447
C	0 8448
C ENTRY POINT FOR LOOPING ON FREQUENCY INCREMENTING.	0 8449
C	0 8450
420 CONTINUE	0 8451
GO TO (370, 450, 500, 610, 660, 1490),KK	0 8452
430 IF (WD(1) .GT. SAVLC) GO TO 450	0 8453
I=I+1	0 8454
GO TO 430	0 8455
450 IF (TABG(J)*SAVLC - TABUP(1)*WD(1)) 460, 490, 490	0 8456
460 W = TABG(J)*SAVLC	0 8457
IF (KTABG .GT. J) GO TO 480	0 8458
SAVLC = SAVLC * 10.00	0 8459
J = 1	0 8460
KK = 2	0 8461
GO TO 830	0 8462
480 J = J+1	0 8463
KK = 2	0 8464
GO TO 830	0 8465
490 J = 1	0 8466
500 IF (J - KTABUP) 520, 510, 530	0 8467
510 W = TABUP(J) * WD(1)	0 8468
KPRINT = 2	0 8469
J = J+1	0 8470
KK = 3	0 8471
GO TO 830	0 8472
520 W = TABUP(J) * WD(1)	0 8473
J = J+1	0 8474
KK = 3	0 8475
GO TO 830	0 8476
530 IF (WD(I+1) .GT. 0.00) GO TO 550	0 8477

C		0 8478
C	THE LAST FREQUENCY IS I,	0 8479
C	MAKE I+1 A DUMMY.	0 8480
C		0 8481
	WD(I+1) = FMAX * TABDN(KTABDN)	0 8482
550	IF (TABUP(JX)*WD(I+1) - WD(I)) 560, 640, 650	0 8483
560	J = JX	0 8484
570	IF (TABUP(J)*WD(I+1) - WD(I)) 580, 590, 600	0 8485
580	J = J+1	0 8486
	GO TO 570	0 8487
590	J = J+1	0 8488
600	I = I+1	0 8489
610	IF ((J-KTABUP) .EQ. 0) KPRINT = 2	0 8490
	IF (J .GT. KTABUP) GO TO 630	0 8491
620	W = TABUP(J)*WD(I)	0 8492
	J = J+1	0 8493
	KK = 4	0 8494
	GO TO 630	0 8495
630	J = 1	0 8496
	GO TO 630	0 8497
640	J = JX+1	0 8498
	GO TO 600	0 8499
650	J = 1	0 8500
660	IF (J .GT. KTABDN) GO TO 690	0 8501
	IF (TABDN(J)*WD(I) - TABUP(JX)*WD(I+1)) 680, 740, 750	0 8502
680	W = TABDN(J) * WD(I)	0 8503
	J = J+1	0 8504
	KK = 5	0 8505
	GO TO 630	0 8506
690	IF (TABUP(I)*WD(I+1) - TABDN(KTABDN)*WD(I)) 700, 700, 760	0 8507
700	J = 1	0 8508
710	IF (TABUP(J)*WD(I+1) - TABDN(KTABDN)*WD(I)) 720, 730, 730	0 8509
720	J = J+1	0 8510
	GO TO 710	0 8511
730	I = I+1	0 8512
	GO TO 610	0 8513
740	J = JX+1	0 8514
	GO TO 600	0 8515
750	J = JX	0 8516
	GO TO 600	0 8517
760	IF (TABDN(KTABDN)*WD(I) - TABG(KTABG)*SAVLU) 770, 810, 820	0 8518
770	J = 1	0 8519
780	IF (TABDN(KTABDN)*WD(I) - TABG(J)*SAVLU) 790, 790, 800	0 8520
790	I = I+1	0 8521
	GO TO 450	0 8522
800	J = J+1	0 8523
	GO TO 780	0 8524
810	SAVLC = SAVLU * 10.00	0 8525
	J = 1	0 8526
	I = I+1	0 8527
	GO TO 450	0 8528
820	SAVLC = SAVLU * 10.00	0 8529
	GO TO 760	0 8530
830	IF (W .GT. FMAX) GO TO 1490	0 8531
	IF (LEQU) WRITE(NDI,100) W,WD(I),SAVLU,I,J,KK,JX,KACT	0 8532
100	FORMAT (' W =',E15.8,' WD(I) =',E15.8,' SAVLU =',E15.8,' I =',I3,	0 8533
	* ' J =',I3,' KK =',I3,' JX =',I3,' KACT =',I3)	0 8534
840	J1 = 1	0 8535
850	IF (NRN) 1210, 910, 850	0 8536
C		0 8537

C	EVALUATE NUMERATOR REAL ROOTS	0 8538
C		0 8539
	860 DO 900 I1=1,NRN	0 8540
	IF (FERN(I1).EQ. 0.00) GO TO 880	0 8541
	870 FNUM(J1) = DCMLPX(1.00,FERN(I1)*W)	0 8542
	GO TO 890	0 8543
	880 FNUM(J1) = DCMLPX(0.00,W)	0 8544
	890 J1 = J1+1	0 8545
	900 CONTINUE	0 8546
	910 IF (NCN) 1220, 940, 920	0 8547
C		0 8548
C	EVALUATE NUMERATOR COMPLEX PAIRS	0 8549
	920 DO 930 I1 = 1,NTOTN,2	0 8550
C		0 8551
C	REAL PART	0 8552
C	IMAGINARY PART	0 8553
C		0 8554
	FNUM(J1) = DCMLPX(1.00 - W**2 / (FBNC(I1+1))**2 ,	0 8555
	1 (2.00* FBNC(I1)*W) / FBNC(I1+1))	0 8556
	J1 = J1+1	0 8557
	930 CONTINUE	0 8558
C		0 8559
C	REPEAT THE ABOVE PROCEDURE FOR DENOMINATOR	0 8560
C		0 8561
	940 J1 = 1	0 8562
	950 IF (NRD) 1230, 1010, 960	0 8563
	960 DO 1000 I1=1,NRD	0 8564
	IF (FBR(I1).EQ. 0.00) GO TO 980	0 8565
	FDEN(J1) = DCMLPX(1.00 , FBR (I1)*W)	0 8566
	GO TO 990	0 8567
	980 FDEN(J1) = DCMLPX(0.00 ,W)	0 8568
	990 J1 = J1+1	0 8569
	1000 CONTINUE	0 8570
	1010 IF (NCD) 1240, 1040, 1020	0 8571
	1020 DO 1030 I1 = 1,NTOTD,2	0 8572
	ALPHA = 1.00 - W**2 / (FBDC(I1+1))**2	0 8573
	BETA = 2.00 * FBDC (I1) * W / FBDC (I1+1)	0 8574
	IF (ALPHA .LT. 1.0-20 .AND. BETA .EQ. 0.00) BETA = 1.0D-10	0 8575
	FDEN(J1) = DCMLPX(ALPHA,BETA)	0 8576
	J1 = J1+1	0 8577
	1030 CONTINUE	0 8578
C		0 8579
C	EVALUATE F(S) WITH COMPLEX ARITHMETIC ROUTINE.	0 8580
C		0 8581
	1040 KN = NRN+NCN	0 8582
	KD = NRD+NCD	0 8583
	PRCD(1) = DCMLPX(1.00,0.00)	0 8584
	IF (KN .LE. 0) GO TO 1090	0 8585
	IF (KD .LE. 0) GO TO 1130	0 8586
	IF (KN .GE.KD) GO TO 1110	0 8587
C		0 8588
C	FACTORS IN DENMINATOR EXCEED THOSE IN NUMERATOR.	0 8589
C		0 8590
	DO 1080 I1=1,KN	0 8591
	PRCD(I1) = FNUM(I1) * PRCD(1)/FDEN(I1)	0 8592
	1080 CONTINUE	0 8593
	1090 K = KN+1	0 8594
	DO 1100 I1=K,KD	0 8595
	PRCD(I1) = PRCD(1)/FDEN(I1)	0 8596
	1100 CONTINUE	0 8597

GO TO 1150	0 8598
C	0 8599
C FACTORS IN NUMERATOR EXCEED THOSE IN DENOMINATOR.	0 8600
C	0 8601
1110 GO 1120 I1=1,K0	0 8602
PRCD(1) = FNUM(I1)*PROD(1)/FDCN(I1)	0 8603
1120 CONTINUE	0 8604
IF (FK .LE. KD) GO TO 1150	0 8605
1130 K = KD+1	0 8606
GO 1140 I1=K,KN	0 8607
PRCD(1) = PRCD(1)*FNUM(I1)	0 8608
1140 CONTINUE	0 8609
1150 PRCD(1) = PRCD(1) * COMPLEX(FK,0.00)	0 8610
C	0 8611
C EVALUATION OF F(S) IS NOW COMPLETE.	0 8612
C NORMAL COMPUTED FORM -- F(jw) = ALPHA + BETA.	0 8613
C ALPHA = REAL PART,	0 8614
C BETA = IMAGINARY PART.	0 8615
C	0 8616
C -----CARTESIAN FORM (X,Y)-----	0 8617
ALPHA = DREAL (PROD(1))	0 8618
BETA = DIMAG (PROD(1))	0 8619
C	0 8620
C IN POLAR FORM -- F(jw) = (AR,PHI).	0 8621
C AR IS AMPLITUDE	0 8622
C PHI IS PHASE ANGLE	0 8623
C	0 8624
C -----POLAR FORM-----	0 8625
RED = (ALPHA**2 + BETA**2)	0 8626
AR = DSQRT(RED)	0 8627
PHI = 0.00	0 8628
IF (AR .NE. 0.00) PHI = DATAN2(BETA,ALPHA)	0 8629
C CONVERT PHI FROM RADIAN TO DEGREES.	0 8630
DPHI = PHI * 57.295800	0 8631
IF (DPHI .LT. 0.00) DPHI = DPHI + 360.00	0 8632
C	0 8633
C -----PRINT FREQUENCY RESPONSE DATA-----	0 8634
C	0 8635
C	0 8636
C SET OUTPUT PARAMETERS	0 8637
C CONVERT AR TO LOG BASE 10 AND DECIBELS.	0 8638
C	0 8639
1250 IF (AR .NE. 0.00) GO TO 1270	0 8640
BELL = -20.00	0 8641
GO TO 1280	0 8642
1270 BELL = 20LOG(AR)	0 8643
1280 DBELL = BELL * 8.6858896100	0 8644
PHI = DPHI / 57.295800	0 8645
W1 = W / 6.283185300	0 8646
C	0 8647
1285 IF (LNCTR = 50) 1320, 1290, 1290	0 8648
1290 CALL PAGENO	0 8649
WRITE (NCT,156) (TITLE(13),13=1,20)	0 8650
156 FORMAT (/10X20A4,/)	0 8651
WRITE (NLT,159)	0 8652
159 FORMAT (1H0,20X,99HFREQ/RAD/SEC FREQ/HERTZ REAL I	0 8653
IMAG AMP DECIBELS RAD DEG /)	0 8654
LNCTR = 1	0 8655
IF (KPRINT .EQ. 1) GO TO 1310	0 8656
KPRINT = 1	0 8657

WRITE (NCT,1620) W,W1,ALPHA,BETA,AR,DBELL,PHI,DPHI	0 8658
1620 FORMAT (17H *****D16.6,D14.6,D15.6,D14.6,D16.6,F10.3,	0 8659
1 F9.4,F10.4,9H *****)	0 8660
GO TO 1330	0 8661
1310 WRITE (NCT,162) W,W1,ALPHA,BETA,AR,DBELL,PHI,DPHI	0 8662
162 FORMAT (9X,D24.6,D14.6,D16.6,D14.6,D16.6,F10.3,F9.4,F10.4)	0 8663
GO TO 1370	0 8664
1320 IF (KPRINT .EQ. 1) GO TO 1350	0 8665
WRITE (NCT,1620) W,W1,ALPHA,BETA,AR,DBELL,PHI,DPHI	0 8666
KPRINT = 1	0 8667
1330 KWCT = 1	0 8668
IF (WD(KWCT+1) .LE. 0.00) GO TO 1370	0 8669
FAT = WD(KWCT) / WD(KWCT+1)	0 8670
JX = 2	0 8671
IF (FAT .LE. 0.4200) GO TO 1370	0 8672
C GET MIDPOINT LOCATION BETWEEN RESONANCE POINTS	0 8673
FAT = (1.+FAT)/2.00	0 8674
GO 1340 KKK=3.30	0 8675
X = FAT - TABUP(KKK)	0 8676
IF (X.LE.0.000) GO TO 1370	0 8677
JX = KKK	0 8678
1340 CONTINUE	0 8679
GO TO 1360	0 8680
1350 WRITE (NCT,162) W,W1,ALPHA,BETA,AR,DBELL,PHI,DPHI	0 8681
1360 LNCNR = LNCNR + 1	0 8682
1370 CONTINUE	0 8683
C	0 8684
C-----SAVE DATA TO PLOT	0 8685
C	0 8686
KSAVE = KSAVE + 1	0 8687
SAVEC(KSAVE) = W	0 8688
SAVEC(KSAVE) = DBELL	0 8689
SAVEF(KSAVE) = DPHI	0 8690
SAVEA(KSAVE) = AR	0 8691
C	0 8692
C	0 8693
1480 CONTINUE	0 8694
C	0 8695
C-----CONTINUE FREQUENCY SWEEP UNTIL LIMITS ARE EXHAUSTED.	0 8696
C	0 8697
IF (W1 .LT. DBLE(FMAX) .AND. KSAVE .LT. KJSAVE) GO TO 420	0 8698
C	0 8699
C NORMAL TERMINATION LOGIC.	0 8700
C	0 8701
1490 CONTINUE	0 8702
C	0 8703
KSAVE = KSAVE-1	0 8704
C	0 8705
RETURN	0 8706
C ERROR EXITS	0 8707
C	0 8708
1210 CALL PAGEHD	0 8709
WRITE (NCT,135) NRN	0 8710
GO TO 1490	0 8711
C	0 8712
1220 CALL PAGEHD	0 8713
WRITE (NCT,137) NCN	0 8714
GO TO 1490	0 8715
C	0 8716
1230 CALL PAGEHD	0 8717

WRITE (NCT,139) NRD	0 8718
GO TO 1450	0 8719
C	0 8720
C	0 8721
C	0 8722
1240 CALL PAGEHD	0 8723
WRITE (NCT,141) NCD	0 8724
135 FORMAT (54HDATA FOR NRN IS INCORRECT IN SUBROUTINE SFREQ2, NRN = ,	0 8725
1 IS)	0 8726
137 FORMAT (54HDATA FOR NCN IS INCORRECT IN SUBROUTINE SFREQ2, NCN = ,	0 8727
1 IS)	0 8728
139 FORMAT (54HDATA FOR NRD IS INCORRECT IN SUBROUTINE SFREQ2, NRD = ,	0 8729
1 IS)	0 8730
141 FORMAT (54HDATA FOR NCD IS INCORRECT IN SUBROUTINE SFREQ2, NCD = ,	0 8731
1 IS)	0 8732
C	0 8733
C	0 8734
STOP	0 8735
END	0 8736
SUBROUTINE RLOCUS (PPR,QQR,SCL,SR,SI,NP,NC,THETA0,	0 8737
1 XMIN,XMAX,YMAX,ALOC)	0 8738
C	0 8739
C	0 8740
IMPLICIT REAL*8(A-H,O-Z)	0 8741
REAL*8 K	0 8742
REAL*4 XSAVE, YSAVE, SAVED, SAVEDP, SAVEDI, SAVEDA	0 8743
C	0 8744
COMPLEX*16 S,P,Q,SC,ERR,DCMPLX	0 8745
INTEGER CUT	0 8746
REAL SINGL,SIGN	0 8747
LOGICAL START	0 8748
C	0 8749
C	0 8750
C	0 8751
C	0 8752
C	0 8753
C	0 8754
C	0 8755
C	0 8756
C	0 8757
C	0 8758
C	0 8759
C	0 8760
C	0 8761
C	0 8762
C	0 8763
C	0 8764
C	0 8765
C	0 8766
C	0 8767
C	0 8768
C	0 8769
DIMENSION PPR(1),QQR(1)	452 8770
DIMENSION PAR(100), DAR(100)	454 8771
DIMENSION XSAVE(500),YSAVE(500)	

```

C                                                     0 8772
COMMON /PSTUFF/                                       0 8773
*      SAVEU(500), SAVEP(500), SAVED(500), SAVEA(500), KSAVE 447 8774
COMMON /LV1 /                                         0 8775
C      V1 (100), V2 (100), V3 (100)                 426 8776
EQUIVALENCE (V1(1),PAR(1)),(V2(1),QAR(1))           0 8777
EQUIVALENCE (SAVEU(1),XSAVE(1)), (SAVEP(1),YSAVE(1)) 0 8778
C                                                     0 8779
DATA OUT, KDSAVE /                                     0 8780
1      6 , 500 /                                     456 8781
C                                                     0 8782
C                                                     0 8783
C                                                     0 8784
C                                                     0 8785
C      KSAVE = 1                                       0 8786
C*****                                               0 8787
C      ALCC SHOULD BE +1 FOR A 100 DEG LOCUS AND -1 FOR A 0 DEG LOCUS. 0 8788
C*****                                               0 8789
F1=3.14159265358979300                               0 8790
ATR=180.00/PI                                         0 8791
C*****                                               0 8792
C      FIND SCALE FACTOR (SCL) IF NOT SPECIFIED.      0 8793
C      IF THE INPUTTED SCL IS POSITIVE, THAT VALUE WILL BE USED TO 0 8794
C      SCALE THE TWO POLYNOMIALS. IF THE SCL IS NEGATIVE, A SCALE 0 8795
C      FACTOR WILL BE ESTIMATED FROM THE SIZE OF THE COEFFICIENTS OF 0 8796
C      THE POLYNOMIALS BY A LEAST SQUARES METHOD.     0 8797
C*****                                               0 8798
IF(SCL.GT.0.00)GO TO 100                             0 8799
CALL FIT(NP,PPR,SLOPEP)                               0 8800
CALL FIT(NQ,QQR,SLOPEQ)                               0 8801
SCL=1.01*(-(NP*SLOPEP+NQ*SLOPEQ)/(NP+NQ))             0 8802
C*****                                               0 8803
C      WRITE HEADINGS                                0 8804
C*****                                               0 8805
100 CONTINUE                                           0 8806
NP1=NP-1                                              0 8807
NQ1=NQ-1                                              0 8808
CALL PAGEHD                                           0 8809
WRITE(OUT,38)                                         0 8810
38 FORMAT('1P(S) =')                                  0 8811
WRITE(OUT,40) PPR(1),(PPR(I+1),I,I=1,NP1)            0 8812
WRITE(OUT,39)                                         0 8813
39 FORMAT('Q(S) =')                                  0 8814
WRITE(OUT,40) QQR(1),(QQR(I+1),I,I=1,NQ1)            0 8815
40 FORMAT (3X,D22.15,7X,3(1X,'+(',D22.15,'*S**',I2,')')/., 0 8816
*      4(1X,'+(',D22.15,'*S**',I2,')'))              0 8817
WRITE(OUT,101) SCL                                    0 8818
101 FORMAT('SCALE FACTOR =',D13.6)                   0 8819
WRITE(OUT,10)SR,SI                                    0 8820
10 FORMAT('STARTING POINT = (',F18.11,') + I(',F18.11,')') 0 8821
WRITE(OUT,15) XMIN,XMAX,YMAX,YMAX                   0 8822
15 FORMAT('SCAN LIMITS:',D13.6,' < X < ',D13.6/' ',12X,'-', 0 8823
*      D12.6,' < Y < ',D13.6)                      0 8824
IF(ALCC)16,16,17                                     0 8825
16 IDEG=0                                             0 8826
GO TO 18                                              0 8827
17 IDEG=180                                          0 8828
18 WRITE(OUT,19) IDEG                                0 8829
19 FORMAT('C',43X,13,' DEGREE LOCUS')               0 8830
NLIN = 0                                             0 8831

```

```

      CALL PAUEND                                0 8832
      WRITE(OUT,20)                              0 8833
20  FORMAT(//13X,'GAIN',33X,'NUCTS',42X,'ERROR'//) 0 8834
C*****0 8835
C  SCALE EQUATIONS BY LETTING S(NEW)=SCL*S(OLD) 0 8836
C*****0 8837
C 0 8838
C  PAR WILL HOLD THE SCALED VERSION OF PPR      0 8839
      ABP=CA3S(PPR(NP))                          0 8840
      DO 120 I=1,NP1                             0 8841
      ABPP=PPR(I)/ABP                             0 8842
      NP1=NP-1                                    0 8843
      DO 110 J=1,NP1                             0 8844
110  ABPP=AJPP/SCL                               0 8845
120  PAR(I)=ABPP*ALOC                             0 8846
      PAR(NP)=ESIGN(1.00,PPR(NP))*ALOC           0 8847
C 0 8848
C  QAR WILL HOLD THE SCALED VERSION OF QQR      0 8849
      ABC=CA3S(QQR(NQ))                          0 8850
      DO 140 I=1,NQ1                             0 8851
      ABQQ=QQR(I)/ABC                             0 8852
      NQ1=NQ-1                                    0 8853
      DO 130 J=1,NQ1                             0 8854
130  ABQQ=AJQQ/SCL                               0 8855
140  QAR(I)=ABQQ                                 0 8856
      QAR(NQ)=ESIGN(1.00,QQR(NQ))               0 8857
C 0 8858
      VK=AEC/AEP                                  0 8859
C  VK=RATIO OF THE MAGNITUDE OF THE LEADING COEFFICIENTS 0 8860
      NQNP=NP-NP                                 0 8861
      DO 150 I=1,NQNP                             0 8862
150  VK=VK*SCL                                   0 8863
C  VK=SCALED RATIO                              0 8864
C*****0 8865
C  INITIALIZE VALUES                          0 8866
C*****0 8867
      ARTLCC=0                                    0 8868
      START=.TRUE.                               0 8869
      SD=DCMPLX(SR/SCL,SI/SCL)                  0 8870
      RMIN=1.0-5                                 0 8871
      RMIN=RMIN/SCL                             0 8872
      RMAX=1.01/SCL                             0 8873
      RD=RMIN*1.01                              0 8874
      RD=RMIN                                    0 8875
      DTM=1.0-8                                  0 8876
      DTPIA=40.00                               0 8877
C  THETA0 IS PASSED INTO THE SUBROUTINE FROM THE MAIN PROGRAM AND 0 8878
C  HOLDS THE ANGLE (IN DEGREES) AT WHICH ALL SEARCHES ARE BEGUN. 0 8879
      THETA1=THETA0                              0 8880
      THMAX=370.00+THETA1                       0 8881
      THETA2=3600.                              0 8882
C  THETA2 IS USED IN THE ADJUST SEARCH RADIUS SECTION. ITS INITIAL 0 8883
C  VALUE IS ARBITRARY.                         0 8884
C*****0 8885
C  START SEARCH                                0 8886
C*****0 8887
190  THETA=THETA1/ATK                             0 8888
      S=SD+DCMPLX(RD*DCOS(THETA),RD*DSIN(THETA)) 0 8889
C  S IS THE POINT TO BE EXAMINED              0 8890
C  NOW EVALUATE P(S) AND Q(S)                 0 8891

```

```

      P=PAR(NP)                                0 8892
      C=CAR(NC)                                0 8893
      DO 200 I=1,NP1                          0 8894
200  P=P*S+PAR(NP-I)                          0 8895
      DO 210 I=1,NQ1                          0 8896
210  C=C*S+CAR(NQ-I)                          0 8897
C      FIND PHASE ANGLE OF P(S)/Q(S)          0 8898
      AP=DREAL(P)                             0 8899
      EP=DIMAG(P)                             0 8900
      CQ=DREAL(Q)                             0 8901
      CQ=DIMAG(Q)                             0 8902
      THN=EP*CC-AP*DD                          0 8903
      THD=AP*CC+EP*DD                          0 8904
C      ARCTAN(THN/THD)=PHASE ANGLE OF P(S)/Q(S) 0 8905
C      CHECK TO SEE IF THIS NEW PHASE ANGLE HAS CROSSED THE 180 DEG LINE. 0 8906
C      FOR BOTH SPEED AND ACCURACY, THIS CHECK IS PERFORMED BY EXAMINING 0 8907
C      THE SIGNS OF THN AND THD RATHER THAN BY USING THE ARCTAN FUNCTION. 0 8908
      ISNNEW=SIGN(1.1,SNGL(THN))              0 8909
      IF(START)GO TO 280                      0 8910
      IF(THD.GE.0.00 .OR. ISNNEW.EQ.ISNCLO)GO TO 281 0 8911
      DTHETA=-C*THETA                         0 8912
      GO TO 281                               0 8913
280  START=.FALSE.                           0 8914
281  ISNCLO=ISNNEW                            0 8915
      IF(DABS(DTHETA).LT.DTM) GO TO 310        0 8916
      IF(THETA1.GT.THMAX)GO TO 300            0 8917
      IF(DTHETA.LT.30.00) DTHETA=DTHETA/2.00 0 8918
      THETA1=THETA1+DTHETA                    0 8919
      GO TO 190                               0 8920
C*****                                     0 8921
C      END SEARCH                            0 8922
C*****                                     0 8923
C      TWO RESULTS OF THE SEARCH ARE POSSIBLE. IF NO ROOT LOCUS POINT 0 8924
C      WAS FOUND, THE COMPUTER GOES TO STATEMENT 300. IF A ROOT LOCUS 0 8925
C      POINT WAS FOUND, THE COMPUTER GOES TO STATEMENT 310. 0 8926
C*****                                     0 8927
C      NO ROOT WAS FOUND.                    0 8928
C*****                                     0 8929
300  RD=RD/1.500                              0 8930
      ANSINC=109.4700                         0 8931
      THETA1=THETA2                            0 8932
      S=50                                     0 8933
      IF(RD.GT.RMIN)GO TO 350                 0 8934
      IF(NFTLOC.EQ.0)GO TO 304                0 8935
      WRITE(OUT,301) RMINN                    0 8936
301  FORMAT('THE LAST POINT PRINTED IS WITHIN ',F7.3,' OF A ROOT.') 0 8937
      RETURN                                  0 8938
304  WRITE(OUT,305) RMINN                     0 8939
305  FORMAT('THE INITIAL POINT IS NOT WITHIN ',F7.3,' OF THE LOCUS.') 0 8940
      RETURN                                  0 8941
C*****                                     0 8942
C      A ROOT LOCUS POINT WAS FOUND.         0 8943
C      EXAMINE SMOOTHNESS OF ROOT LOCUS CURVE. SEE IF THIS POINT WILL BE 0 8944
C      USED. IF THE CURVE BENDS TOO SHARPLY, POINT WILL NOT BE ACCEPTED, 0 8945
C      THE PROGRAM WILL START AT THE OLD SC, SHRINK THE SEARCH RADIUS RD, 0 8946
C      AND LOOK FOR A CLOSER POINT WHICH IS MORE IN LINE WITH THE CURVE. 0 8947
C*****                                     0 8948
310  CONTINUE                                0 8949
C      FIND ACUTE ANGLE BETWEEN THETA1 AND THETA2 (THDIF) 0 8950
      THDIF=DABS(THETA1-THETA2)               0 8951

```

	THDIF=DMIN1(THDIF1,360.00-THDIF1)	0 8952
C	ADJUST SEARCH RADIUS IF THDIF IS LESS THAN 15 DEG OR	0 8953
C	GREATER THAN 30 DEG	0 8954
	IANG=THDIF/15.00	0 8955
	IF(IANG-1) 320,340,330	0 8956
C		0 8957
C	GO HERE IF THDIF .LT. 15. DEG	0 8958
320	RD=1.500*RD	0 8959
	ANGINC=138.5900	0 8960
	IF(RC.LE.RMAX)GO TO 345	0 8961
	RD=RMAX	0 8962
C		0 8963
C	GO HERE IF 15. .GE. THDIF . LT. 30. DEG	0 8964
340	ANGINC=120.00	0 8965
C		0 8966
C	THIS STATEMENT IS REACHED ONLY IF THE ROOT LOCUS POINT IS ACCEPTED	0 8967
345	CONTINUE	0 8968
	NRTLCC=NRTLCC+1	0 8969
	CQCP=COAES(Q)/COABS(P)	0 8970
	K=VK*CQCP*ALCC	0 8971
	X=CREAL(S)*SCL	0 8972
	Y=CIMAG(S)*SCL	0 8973
	YY=DABS(Y)	0 8974
C	K = OPEN LOOP GAIN	0 8975
C	X = REAL COORDINATE OF ROOT ASSOCIATED WITH K	0 8976
C	Y = IMAGINARY COORDINATE OF ROOT ASSOCIATED WITH K	0 8977
	ERF=G+P*CQCP	0 8978
	IF(DABS(THN/THD).LE.2.0-S)GO TO 271	0 8979
	ANGL=ATR*DATAN2(THN,THD)	0 8980
	WRITE(OUT,270) ANGL	0 8981
270	FORMAT(' MINIMUM VALUE OF DTHETA REACHED, PRESENT VALUE OF THETA I	0 8982
	*S',F18.9)	0 8983
271	NLINE = NLINE + 1	0 8984
	IF (NLINE .LT. 50) GO TO 314	0 8985
	NLINE = 1	0 8986
	CALL PAGEHD	0 8987
	WRITE (OUT,20)	0 8988
314	WRITE(OUT,50)K,X,Y,ERR	0 8989
	YSAVE(KSAVE) = Y	0 8990
	XSAVE(KSAVE) = X	0 8991
50	FORMAT(S(6X,G18.9))	0 8992
	IF ((X .LT. XMIN .OR. X .GT. XMAX .OR. YY .GT. YMAX)	0 8993
	* .OR. (KSAVE .GE. KSAVE)) RETURN	0 8994
	KSAVE = KSAVE + 1	0 8995
	GO TO 350	0 8996
C		0 8997
C	GO HERE IF THDIF .GE. 30. DEG	0 8998
330	CONTINUE	0 8999
	IF(NRTLCC.EQ.1)GO TO 340	0 9000
	IF(RC.LE.RMIN)GO TO 340	0 9001
	RD=RD/1.500	0 9002
	ANGINC=109.4700	0 9003
	THETA1=THETA2	0 9004
	S=SU	0 9005
C	*****	0 9006
C	PREPARE TO LOOK FOR NEXT POINT ON ROOT LOCUS	0 9007
C	THIS SECTION IS REACHED ONLY IF A ROOT LOCUS POINT WAS FOUND.	0 9008
C	*****	0 9009
350	THETA2=THETA1	0 9010
	SO=S	0 9011

	THETA1=THETA2-ANGINC		0 9012
	THMAX =THETA2+ANGINC		0 9013
	DTHEA=4.01		0 9014
	START=.TRUE.		0 9015
	CG TC 190		0 9016
	END		0 9017
/			
	SUBROUTINE FIT(N,ARRAY,SLOPE)		0 9018
C		DEBUG # 75	0 9019
	REAL *8 Y,SUMY,SUMIY,DLOG10,SLOPE,ARRAY(100)		0 9020
C			0 9021
	N=C		0 9022
	ISUM=0		0 9023
	ISQSUM=0		0 9024
	SUMY=0.00		0 9025
	SUMIY=0.00		0 9026
	DO 4 I=1,N		0 9027
	IF (ARRAY(I)) 1,4,2		0 9028
	1 Y=DLOG10(-ARRAY(I))		0 9029
	GO TC 3		0 9030
	2 Y=DLOG10(ARRAY(I))		0 9031
	3 N=M+1		0 9032
	ISUM=ISUM+1		0 9033
	ISQSUM=ISQSUM+I*I		0 9034
	SUMY=SUMY+Y		0 9035
	SUMIY=SUMIY+I*Y		0 9036
	4 CONTINUE		0 9037
	SLOPE=(M*SUMIY-ISUM*SUMY)/(M*ISQSUM-ISUM*ISUM)		0 9038
	RETURN		0 9039
	END		0 9040
/			
	FUNCTION DIMAG (Y)		0 9041
C		DEBUG # 76	0 9042
	COMPLEX*16 Y		0 9043
	COMPLEX SY		0 9044
	SY = Y		0 9045
	DIMAG = DBLE (AIMAG(SY))		0 9046
	RETURN		0 9047
	END		0 9048
/			
	FUNCTION DREAL (Y)		0 9049
C		DEBUG # 77	0 9050
	COMPLEX*16 Y		0 9051
	COMPLEX SY		0 9052
	SY = Y		0 9053

```

CREAL = CDBE (REAL(SY))      0 9054
RETURN                        0 9055
END                            0 9056

```

```

SUBROUTINE CANCEC (R)      0 9057
C                               DEBUG # 78      0 9058
C      IMPLICIT REAL*8(A-H,O-Z)      0 9059
C THIS ROUTINE CANCELS OUT THE ZERO, REAL, AND THE COMPLEX ROOTS THAT AR      0 9060
C COMMON TO THE NUMERATOR AND DENOMINATOR OF THE TRANSFER FUNCTION R.      0 9061
C      0 9062
C --- R(1) = NUMBER OF REAL ROOTS IN THE NUMERATOR      0 9063
C --- R(2) = NUMBER OF COMPLEX PAIRS IN THE NUMERATOR      0 9064
C --- R(3) = NUMBER OF ZERO ROOTS IN THE NUMERATOR      0 9065
C --- R(4) = NUMBER OF REAL ROOTS IN THE DENOMINATOR      0 9066
C --- R(5) = NUMBER OF COMPLEX PAIRS IN THE DENOMINATOR      0 9067
C --- R(6) = NUMBER OF ZERO ROOTS IN THE DENOMINATOR      0 9068
C --- R(7) = GAIN      0 9069
C --- R(8)..R(11) = NUMERATOR REAL ROOTS ARRAY      0 9070
C --- R(11+1)..R(J) = NUMERATOR COMPLEX PAIRS ARRAY      0 9071
C --- R(J+1)..R(K) = DENOMINATOR REAL ROOTS ARRAY      0 9072
C --- R(K+1)..R(L) = DENOMINATOR COMPLEX PAIRS ARRAY      0 9073
C      0 9074
C      DIMENSION R(1)      0 9075
C      NR=R(1)+.000100      0 9076
C      NCP=R(2)+.000100      0 9077
C      MR=R(4)+.000100      0 9078
C      MCP=R(5)+.000100      0 9079
C      KK=7+NR+MR+2*(NCP+MCP)      0 9080
C      N = 7 + NR      0 9081
C      IF ((NR.EQ.0).OR.(MR.EQ.0)) GO TO 100      0 9082
C      J=8+NR+2*NCP      0 9083
C      K=J-1+MR      0 9084
C      NN=0      0 9085
C      DO 140 I=8,N      0 9086
C      II=I-NN      0 9087
C      DO 100 JJ=J,K      0 9088
C      JJJ=JJ      0 9089
C      IF (ABS(R(II)/R(JJJ)-1.00).LT.1.00-7) GO TO 110      0 9090
100 CONTINUE      0 9091
C      GO TO 140      0 9092
110 DO 130 L=11,KK      0 9093
C      IF (L.GE.(JJJ-1)) GO TO 120      0 9094
C      R(L)=R(L+1)      0 9095
C      GO TO 130      0 9096
120 R(L)=R(L+2)      0 9097
130 CONTINUE      0 9098
C      KK=KK-2      0 9099
C      J=J-1      0 9100
C      K=K-2      0 9101
C      NN=NN+1      0 9102
C      IF ((NN.LC.NR).OR.(NN.EQ.MR)) GO TO 150      0 9103
140 CONTINUE      0 9104
150 R(1)=R(1)-DFLOAT(NN)      0 9105
C      R(4)=R(4)-DFLOAT(NN)      0 9106
C      NR=NR-NN      0 9107

```


MR=MR-NN	0 9108
160 IF ((NCP.EQ.0).OR.(MCP.EQ.0)) GO TO 230	0 9109
NNN=E+NR	0 9110
N=NNN-1+2*NCP	0 9111
J=N+1+MR	0 9112
K=J-1+2*MCP	0 9113
NN=0	0 9114
DO 210 I=NNN,N,2	0 9115
II=I-2*NN	0 9116
DO 170 JJ=J,K,2	0 9117
JJJ=JJ	0 9118
IF ((R(II),EQ.P(JJJ)).AND.(R(II+1),EQ.R(JJJ+1))) GO TO 180	0 9119
170 CONTINUE	0 9120
GO TO 210	0 9121
180 DO 200 L=II,KK	0 9122
IF (L.GE.(JJJ-2)) GO TO 190	0 9123
R(L)=R(L+2)	0 9124
GO TO 200	0 9125
190 R(L)=R(L+4)	0 9126
200 CONTINUE	0 9127
KK=KK-4	0 9128
J=J-2	0 9129
K=K-4	0 9130
NN=NN+1	0 9131
IF ((NN.EQ.NCP).OR.(NN.EQ.MCP)) GO TO 220	0 9132
210 CONTINUE	0 9133
220 R(2)=R(2)-DFLOAT(NN)	0 9134
R(5)=R(5)-DFLOAT(NN)	0 9135
230 X=DMIN1(R(3),R(6))	0 9136
R(3)=R(3)-X	0 9137
R(6)=R(6)-X	0 9138
RETURN	0 9139
END	0 9140

SUBROUTINE FORMB (KR, KC, RLRT, CMPR, FBR, FBC)	0 9141
C	0 9142
IMPLICIT REAL*8(A-H,O-Z)	0 9143
CFORMB FACTORED FORM TIME CONSTANTS, DAMPING AND FREQUENCY	0 9144
C CALLING SEQUENCE FOR SUBROUTINE FORMB IS AS FOLLOWS	0 9145
C CALL FORMB (KR, KC, RLRT, CMPR, FBR, FBC)	0 9146
C KR -- COUNT OF REAL ROOTS, MAY OR MAY NOT INCLUDE ZEROS.	0 9147
C KC -- COUNT OF THE COMPLEX PAIRS OF ROOTS.	0 9148
C RLRT -- STORAGE BLOCK CONTAINING ALL REAL ROOTS.	0 9149
C CMPR -- STORAGE BLOCK CONTAINING COMPLEX PAIRS OF ROOTS	0 9150
C FBC -- FORM B COMPLEX PAIR BLOCK (OUTPUT FROM ROUTINE)	0 9151
C FBR -- FORM B REAL ROOT BLOCK (OUTPUT FROM ROUTINE)	0 9152
C	0 9153
DIMENSION RLRT(1),CMPR(1),FBR(1),FBC(1)	0 9154
C	0 9155
IF (KR) 140, 140, 100	0 9156
100 DO 130 L = 1,KR	0 9157
IF (RLRT(L)) 120, 110, 120	0 9158
110 FBR(L) = 0.0	0 9159
GO TO 130	0 9160
120 FBR(L) = -1.00/ RLRT(L)	0 9161

130 CONTINUE	0 9162
140 IF (KC) 170, 170, 150	0 9163
150 KK = 2*KC	0 9164
DO 160 L = 2, KK, 2	0 9165
FBC(L) = USQRT(CMPR(L-1)**2 + CMPR(L)**2)	0 9166
160 FBC(L-1) = -CMPR(L-1)/FBC(L)	0 9167
170 RETURN	0 9168
END	0 9169
SUERROUTINE DFORMB(KR,KC,RLRT,CMPR,FBR,FBC,SCMPR,SF,ACCD,GB)	0 9170
C	0 9171
C	0 9172
IMPLICIT REAL*8(A-H,O-Z)	0 9173
COMMON FACTORED FORM TIME CONSTANTS, DAMPING AND FREQUENCY	0 9174
DIMENSION RLRT(1),CMPR(1),FBR(1),FBC(1),SCMPR(1)	0 9175
DIMENSION GB(2)	0 9176
LOGICAL DEBUG,LEQU	0 9177
COMMON /LDEBUG/ DEBUG(120)	0 9178
EQUIVALENCE (LEQU,DEBUG(80))	0 9179
C KR = NUMBER OF REAL AND ZERO ROOTS IN RLRT	0 9180
C KC = NUMBER OF COMPLEX PAIRS IN CMPR	0 9181
C KK = 2*KC	0 9182
C RLRT = REAL AND ZERO ROOT ARRAY	0 9183
C CMPR = COMPLEX PAIR ARRAY	0 9184
C SF = 1.0	0 9185
C ACCD = GAIN FOR NUMERATOR	0 9186
C ACCD = 1.0 FOR DENOMINATOR	0 9187
GB(1)=ACCD	0 9188
LZ=0	0 9189
SCI=1.00/SF	0 9190
IF(KR)140, 140, 100	0 9191
100 DO 130 L=1,KR	0 9192
IF(RLRT(L))120, 110, 120	0 9193
110 FBR(L)=0.00	0 9194
C COUNT ZERO ROOTS	0 9195
LZ=LZ+1	0 9196
GO TO 130	0 9197
C PER = TIME CONSTANTS ASSOCIATED WITH REAL ROOTS	0 9198
120 FBR(L)=-1.00/(RLRT(L)*SCI)	0 9199
C COMPUTE PRODUCT OF REAL ROOTS	0 9200
GB(1)=-GB(1)*RLRT(L)	0 9201
130 CONTINUE	0 9202
140 IF(KC)170, 170, 150	0 9203
C CYCLE THROUGH ALL COMPLEX PAIRS	0 9204
150 KK=2*KC	0 9205
DO 160 L=2, KK, 2	0 9206
C CONTINUE GETTING PRODUCT OF ALL ROOTS	0 9207
GB(1)=GB(1)*(CMPR(L-1)**2+CMR(L)**2)	0 9208
SCMPR(L-1)=CMR(L-1)*SCI	0 9209
C FBC(L) = FREQUENCY ASSOCIATED WITH COMPLEX PAIR	0 9210
C FBC(L-1) = DAMPING ZETA ASSOCIATED WITH COMPLEX PAIR	0 9211
C FORM:	0 9212
C $S^2 + 2*\zeta\omega_n*S + \omega_n^2 = 0$	0 9213
C FBC(L) = ω_n	0 9214
C FBC(L-1) = ζ	0 9215

```

      FBC(L)=DSQRT(SCMPR(L-1)**2+(CMPR(L)*SCI)**2)      0 9216
160 FBC(L-1)= -SCMPR(L-1)/FBC(L)                      0 9217
C      RETURN WITH PRODUCT OF ALL NCN-ZERO ROOTS IN GB(1) AND GB(2) 0 9218
170 GB(2)=GE(1)                                       0 9219
      IF (LZ .EQ. 0) GO TO 160                        0 9220
      GB(2)=GB(2)*SF*LZ                               0 9221
180 CONTINUE                                          0 9222
      IF (.NOT. LEQU) RETURN                          0 9223
      CALL WRITE( RLRT,1,KR,6H RLRT ,1)              C 9224
      CALL WRITE( CMPR,1,KK,6H CMPR ,1)              0 9225
      CALL WRITE( FBR ,1,KR,6H FBR ,1)               0 9226
      CALL WRITE( FBC ,1,KK,6H FBC ,1)               0 9227
      CALL WRITE( ACCD,1, 1,6H ACCD ,1)              0 9228
      CALL WRITE( GB,1, 2,6H GB ,1)                  0 9229
      RETURN                                          0 9230
      END                                             0 9231

      SUBROUTINE RTCP (RTS,POLY,TEMP,KPLY)              0 9232
C                                                     0 9233
C                                                     DEBUG # 81
      IMPLICIT REAL*8(A-H,O-Z)                        0 9234
      CRTOP      TRANSFER FUNCTION ROOTS TO TRANSFER FUNCTION POLYNOMIALS 0 9235
C                                                     0 9236
C      SUBROUTINE RTCP,  ROOTS CONVERTED TO POLYNOMIAL 0 9237
C                                                     0 9238
C --- RTS(1) = NUMBER OF REAL ROOTS IN THE NUMERATOR 0 9239
C --- RTS(2) = NUMBER OF COMPLEX PAIRS IN THE NUMERATOR 0 9240
C --- RTS(3) = NUMBER OF ZERO ROOTS IN THE NUMERATOR 0 9241
C --- RTS(4) = NUMBER OF REAL ROOTS IN THE DENOMINATOR 0 9242
C --- RTS(5) = NUMBER OF COMPLEX PAIRS IN THE DENOMINATOR 0 9243
C --- RTS(6) = NUMBER OF ZERO ROOTS IN THE DENOMINATOR 0 9244
C --- RTS(7) = GAIN FACTOR 0 9245
C --- RTS(8)...RTS(1) = NUMERATOR REAL ROOTS ARRAY 0 9246
C --- RTS(1+1)...RTS(J) = NUMERATOR COMPLEX ROOTS ARRAY 0 9247
C --- RTS(J+1)...RTS(K) = DENOMINATOR REAL ROOTS ARRAY 0 9248
C --- RTS(K+1)...RTS(L) = DENOMINATOR COMPLEX ROOTS ARRAY 0 9249
C --- POLY(1) = DEGREE OF THE NUMERATOR 0 9250
C --- POLY(2) = DEGREE OF THE DENOMINATOR 0 9251
C --- POLY(3)...POLY(1) = ALL COEFFICIENTS OF NUMERATOR FOR ASCENDING PD 0 9252
C --- POLY(1+1)...POLY(J) = ALL COEFFICIENTS OF DENOMINATOR FOR ASCENDIN 0 9253
C --- OF S 0 9254
C --- TEMP = TEMPORARY STORAGE 0 9255
C --- NCD = NUMBER OF COMPLEX PAIRS IN DENOMINATOR 0 9256
C --- NCN = NUMBER OF COMPLEX PAIRS IN NUMERATOR 0 9257
C --- NDEN = TOTAL NUMBER OF DENOMINATOR ROOTS IN RTS ARRAY 0 9258
C --- NNUM = TOTAL NUMBER OF NUMERATOR ROOTS IN RTS ARRAY 0 9259
C --- NRD = NUMBER OF REAL ROOTS IN THE DENOMINATOR 0 9260
C --- NRN = NUMBER OF REAL ROOTS IN THE NUMERATOR 0 9261
C --- NZD = NUMBER OF ZERO ROOTS IN THE DENOMINATOR 0 9262
C --- NZN = NUMBER OF ZERO ROOTS IN THE NUMERATOR 0 9263
C --- KPLY = DIMENSION SIZE OF POLY IN CALLING PROGRAM. 0 9264
C                                                     0 9265
      DIMENSION RTS(1),POLY(1),TEMP(1)              0 9266
      KTAPE = 6                                       0 9267
      NRN=RTS(1) + 0.100                             0 9268
      NCN=RTS(2) + 0.100                             0 9269

```

NZN=RTS(3) + 0.100	0 9270
NRD=RTS(4) + 0.100	0 9271
NCD=RTS(5) + 0.100	0 9272
NZD=RTS(6) + 0.100	0 9273
DO 100 I = 1,KPLY	0 9274
FCLY(I)=0.00	0 9275
100 TEMP(I) = 0.00	0 9276
C	0 9277
NUM = 2 * NCA + NRN + NZN	0 9278
NDEN = 2 * NCD + NRJ + NZD	0 9279
FOLY(1)=NUM	0 9280
FOLY(2)=NDEN	0 9281
C	0 9282
KP = 0	0 9283
FOLY (3) = 1.00	0 9284
IF (NRN) 510, 110, 130	0 9285
110 IF (NCN) 510, 120, 190	0 9286
C NUMERATOR IS GAIN TERM ONLY	0 9287
120 KP = NZN + J	0 9288
FOLY(3) = 0.00	0 9289
FOLY(KP) = RTS(7)	0 9290
KP = KP+1	0 9291
GO TO 290	0 9292
C NUMERATOR REAL ROOTS	0 9293
130 TEMP (2) = RTS(8)	0 9294
FOLY (4) = TEMP (2)	0 9295
IF (NRN-1) 180, 180, 140	0 9296
140 DO 170 K = 2, NRN	0 9297
DO 150 K1 = 1, K	0 9298
150 TEMP (K1+1) = RTS(K+7) * POLY (K1+2) + POLY (K1+3)	0 9299
DO 160 K2 = 1, K	0 9300
160 FOLY (K2+3) = TEMP (K2+1)	0 9301
170 CONTINUE	0 9302
C	0 9303
180 IF (NCN) 510, 250, 190	0 9304
C	0 9305
C INCLUDE THE NUMERATOR COMPLEX ROOTS	0 9306
190 KNR = NRN	0 9307
KC = NRN +8	0 9308
KCM = 2 * NCN + KC - 1	0 9309
DO 240 L = KC, KCM, 2	0 9310
TEM1 = 2.00*RTS(L)/RTS(L+1)	0 9311
TEM2 = 1.00/RTS(L+1)**2	0 9312
LL = L-6	0 9313
DO 220 L2 = 1, LL	0 9314
TEM3 = 0.00	0 9315
IF (L2-1) 210, 210, 200	0 9316
200 TEM3 = TEM2 * POLY (L2+1)	0 9317
210 TEMP(L2+1) = POLY (L2+3) + TEM1 * POLY (L2+2) + TEM3	0 9318
220 CONTINUE	0 9319
KNR = KNR + 2	0 9320
DO 230 L3 = 1, KNR	0 9321
230 FOLY (L3+3) = TEMP (L3+1)	0 9322
240 CONTINUE	0 9323
C	0 9324
C ENTER GAIN FACTOR, ZERO ROOTS, RESTORE COEFFICIENTS	0 9325
250 KP = NZN + 4	0 9326
KS = 2 * NCN + NRN	0 9327
DO 260 J = 1, KS, 1	0 9328
FOLY (KP) = TEMP (J+1) * RTS (7)	0 9329

260 KP = KP+1	0 9330
POLY (NZN+3) = 1.00* RTS (7)	C 9331
IF (NZN) 290, 290, 270	0 9332
270 DO 280 J = 1, NZN, 1	0 9333
280 POLY (J+2) = 0.00	0 9334
C	0 9335
C PROCESS DENOMINATOR ROOTS, KP IS LOCATION FOR STORING FIRST	0 9336
C DENOMINATOR COEFFICIENT	0 9337
C	0 9338
290 POLY (KP) = 1.00	0 9339
IF (NRD) 310, 300, 340	C 9340
300 IF (NCD) 310, 310, 400	0 9341
C DENOMINATOR REAL ROOTS (KR IS LOCATION FOR FIRST ROOT)	0 9342
310 IF (NZD) 310, 300, 320	0 9343
320 KRIP = KP + NZD	0 9344
DO 330 IS = KP, KRIP	C 9345
POLY (IS) = 0.00	0 9346
330 CONTINUE	0 9347
POLY (KRIP) = 1.00	C 9348
GO TO 300	0 9349
340 KR = 2*NCN + NRN + 8	0 9350
POLY (KP+1) = RTS (KR)	0 9351
TEMP (2) = RTS (KR)	0 9352
IF (NRD-1) 390, 390, 350	0 9353
DO 360 K=2, NRD	0 9354
NC1 = KR+K-1	0 9355
DO 360 K1 = 1, K	0 9356
NC2 = KP+K1-1	0 9357
360 TEMP (K1+1) = RTS (NC1)*POLY (NC2) + POLY (NC2+1)	0 9358
DO 370 K2 = 1, K	0 9359
NC3 = KP+K2	0 9360
370 POLY (NC3) = TEMP (K2+1)	0 9361
380 CONTINUE	C 9362
C	0 9363
390 IF (NCD) 310, 460, 400	C 9364
C	0 9365
C PROCESS DENOMINATOR COMPLEX ROOTS	0 9366
400 KDR = NRD	0 9367
KC = 2*NCN + NRN + NRD + 6	0 9368
KCM = 2* NCD + KC-1	C 9369
DO 450 L = KC, KCM, 2	0 9370
TEM1 = 2.00*RTS (L)/RTS (L+1)	0 9371
TEM2 = 1.00/RTS (L+1)**2	0 9372
LL = L-(2*NCN+NRN+6)	0 9373
DO 430 L2 = 1, LL	C 9374
NC5 = KP+L2-1	0 9375
TEM3 = 0.00	0 9376
IF (L2-1) 420, 420, 410	0 9377
410 TEM3 = TEM2 * POLY (NC5-1)	0 9378
420 TEMP (L2+1) = POLY (NC5+1) + TEM1 * POLY (NC5) + TEM3	0 9379
430 CONTINUE	C 9380
KDR = KDR + 2	0 9381
DO 440 L3 = 1, KDR	0 9382
NC6 = KP+L3	0 9383
440 POLY (NC6) = TEMP (L3+1)	C 9384
450 CONTINUE	C 9385
C	0 9386
460 KD = KP + NZD + 1	0 9387
KS = 2 * NCD + NRD	C 9388
DO 470 M = 1, KS, 1	0 9389

	POLY (KD) = TEMP (M+1)	0 9390
470	KD = KJ+1	0 9391
	KD = KP+NZD-1	0 9392
	POLY (KD+1) = POLY (KP)	0 9393
	IF (NZD) 500, 500, 480	0 9394
480	CO 490 J = KP, KD, 1	0 9395
490	POLY (J) = 0.00	0 9396
C		0 9397
	500 RETURN	0 9398
C	ERROR COMMENT AND RETURN TO MAIN PROGRAM	0 9399
	510 WRITE (KTAPE,1002)	0 9400
10020	FORMAT (20H1 A NEGATIVE COUNT OF ROOTS WAS ENCOUNTERED I RTOP. PD	0 9401
	ILYNCHIAL COULD NOT BE OBTAINED.)	0 9402
	RETURN	0 9403
	END	0 9404
	SUBROUTINE TRFB (ND,RX,KR,KC,KZ,FBR,FBC,GG,ZGV,KSIZE)	0 9405
C		0 9406
C	DEBUG # 82	0 9407
	IMPLICIT REAL*8(A-H,O-Z)	0 9408
CTRFB	TRANSFER INPUT ROOTS FORMB AND FORMC	0 9409
C	ND -- IF INPUT (0) WE HAVE NUMERATOR, IF INPUT (1) A DENOMINATOR	0 9410
C	RX -- ENTIRE BLOCK (COUNTS,GAIN,ROOTS)	0 9411
C	KR -- RUNNING COUNT OF ACCUMULATED REALS FOR ANY GIVEN CASE	0 9412
C	KC -- SAME AS ABOVE BUT FOR COMPLEX	0 9413
C	KZ -- COUNT OF ACCUMULATED ZEROS FOR ANY GIVEN CASE	0 9414
C	FBR -- FORM (B) REAL STORAGE BLOCK	0 9415
C	FBC -- FORM (B) COMPLEX STORAGE BLOCK	0 9416
C	GG -- RUNNING GAIN TERM	0 9417
C	ZGV -- IF OUTPUT OTHER THAN ZERO, ABSF(ZETA) EXCEEDED (1)	0 9418
C	KSIZE = DIMENSIONED SIZE OF FBR AND FBC .	0 9419
C	IT IS ASSUMED THAT THE COUNT FOR THE ACCUMULATED ROOTS WILL BE ZEROED	0 9420
C	OUT AT THE BEGINNING OF EACH CASE.	0 9421
C	ANOTHER TASK IN THE (MAIN) PROGRAM IS CHECKING THE (ZETA) FLAG.	0 9422
	DIMENSION RX(1),FBR(1),FBC(1)	0 9423
	IF (GG .EQ. 0.00) RETURN	0 9424
	IF (ND.GT.0) GO TO 90	0 9425
	IF (RX(7) .EQ. 0.00) GO TO 200	0 9426
	GG=GG*RX(7)	0 9427
90	KRX1 = RX(1) + 0.100	0 9428
	KRX2 = RX(2) + 0.100	0 9429
	KRX3 = RX(3) + 0.100	0 9430
	KRX4 = RX(4) + 0.100	0 9431
	KRX5 = RX(5) + 0.100	0 9432
	KRX6 = RX(6) + 0.100	0 9433
	IF (ND) 100, 100, 110	0 9434
100	J = 7	0 9435
	JCR = KRX1	0 9436
	JCC = 2*KRX2	0 9437
	JCZ = KRX3	0 9438
	CO TO 120	0 9439
110	J = 7 + KRX1 + 2 * KRX2	0 9440
	JCR = KRX4	0 9441
	JCC = 2*KRX5	0 9442
	JCZ = KRX6	0 9443

120 IF (JCR) 150, 150, 130	0 9444
130 DO 140 M = 1, JCR	0 9445
KR = KR+1	0 9446
L = J+M	0 9447
FBR(KR) = RX(L)	0 9448
140 CONTINUE	0 9449
150 IF (JCC) 190, 190, 160	0 9450
160 DO 180 M = 2, JCC, 2	0 9451
KC = KC+1	0 9452
KK = 2*KC	0 9453
L = J+JCR+M	0 9454
FBC(KK-1) = RX(L-1)	0 9455
FBC(KK) = RX(L)	0 9456
IF (CABS(RX(L-1))- 1.00) 180, 180, 170	0 9457
170 ZDV = 1.00	0 9458
180 CONTINUE	0 9459
190 KZ = KZ+J CZ	0 9460
RETURN	0 9461
200 GG=0.00	0 9462
KR=0	0 9463
KC=0	0 9464
KZ=0	0 9465
DO 210 I=1, KSIZE	0 9466
FBR(I)=0.00	0 9467
210 FBC(I)=0.00	0 9468
RETURN	0 9469
END	0 9470
SUBROUTINE ITFF (NR, NC, NZ, KR, KC, KZ, G, RN, CN, RD, CD, R, KSIZE)	0 9471
C	0 9472
C	0 9473
IMPLICIT REAL*8(A-H, O-Z)	0 9474
C	0 9475
C --- NR = NUMBER OF REAL ROOTS IN THE NUMERATOR	0 9476
C --- NC = NUMBER OF COMPLEX PAIRS IN THE NUMERATOR	0 9477
C --- NZ = NUMBER OF ZERO ROOTS IN THE NUMERATOR	0 9478
C --- KR = NUMBER OF REAL ROOTS IN THE DENOMINATOR	0 9479
C --- KC = NUMBER OF COMPLEX PAIRS IN THE DENOMINATOR	0 9480
C --- KZ = NUMBER OF ZERO ROOTS IN THE DENOMINATOR	0 9481
C --- G = GAIN	0 9482
C --- RN = NUMERATOR REAL ROOT ARRAY	0 9483
C --- CN = NUMERATOR COMPLEX PAIRS ARRAY	0 9484
C --- RD = DENOMINATOR REAL ROOT ARRAY	0 9485
C --- CD = DENOMINATOR COMPLEX PAIRS ARRAY	0 9486
C --- R = ARRAY CONTAINING NUMBER OF ROOTS AND ROOT ARRAYS.	0 9487
C -KSIZE= DIMENSIONED SIZE OF R IN CALLING PROGRAM.	0 9488
C	0 9489
DIMENSION RN(1), CN(1), RD(1), CD(1), R(1)	0 9490
LOGICAL DEBUG, LEQU	0 9491
DATA NOT/6/	0 9492
COMMON /LDEBUG/ DEBUG(120)	0 9493
EQUIVALENCE (LEQU, DEBUG(85))	0 9494
IF (G.EQ.0.00) GO TO 80	0 9495
R(1)=NR	0 9496
R(2)=NC	0 9497

K(3)=NZ	0 9498
K(4)=KR	0 9499
K(5)=KC	0 9500
K(6)=KZ	0 9501
K(7)=G	0 9502
L=7+NR	0 9503
IF (NR.EC.0.DC) GO TO 20	0 9504
DO 10 I=8,L	0 9505
10 R(I)=RN(I-7)	0 9506
20 M=L+1	0 9507
L=L+2*NC	0 9508
IF(NC.LE.0) GO TO 40	0 9509
DO 30 I=M,L	0 9510
J=I-M+1	0 9511
30 R(I)=CN(J)	0 9512
40 M=L+1	0 9513
L=L+KR	0 9514
IF(KR.LE.0) GO TO 60	0 9515
DO 50 I=M,L	0 9516
J=I-M+1	0 9517
50 R(I)=RD(J)	0 9518
60 IF(KC.LE.0) GO TO 100	0 9519
M=L+1	0 9520
L=L+2*KC	0 9521
DO 70 I=M,L	0 9522
J=I-M+1	0 9523
70 R(I)=C(J)	0 9524
DO 70 I=1,KSIZE	0 9525
80 DO 90 I=1,KSIZE	0 9526
90 R(I)=0.000	0 9527
100 CONTINUE	0 9528
IF(.NOT.LEQU) RETURN	0 9529
WRITE(NCT,1000)	0 9530
1000 FORMAT (' SUBROUTINE TTF ROOTS ARRAY IS '	0 9531
CALL WRITES(R,1,KSIZE,1)	0 9532
RETURN	0 9533
END	0 9534

SUBROUTINE QDRVR (A,N,KR,RI,KR)	0 9535
C	0 9536
IMPLICIT REAL*8(A-H,O-Z)	0 9537
C	0 9538
C ---LATEST MOD ON 09/04/75	0 9539
C ---WHEN ENTIRE PACKAGE WAS MADE INTO G-PRECISION	0 9540
C	0 9541
CC FOR EXTENDED PRECISION, Q, VI MUST BE DECLARED REAL*16.	0 9542
C ALL OTHERS DECLARED REAL*8.	0 9543
C	0 9544
C	0 9545
DIMENSION A(KR,1), RR(1), RI(1)	0 9546
DIMENSION G(100,100), VI(100)	410 9547
C	0 9548
COMMON /LWOKK1/	0 9549
F W1(100,100), W2(100,100)	432 9550
C	0 9551

EQUIVALENCE (Q(1),W1(1))	0 9552
LOGICAL DEBUG,LEQU	0 9553
COMMON /LDEBUG/ DEBUG(120)	0 9554
EQUIVALENCE (LEQU,DEBUG(84))	0 9555
DATA NOT/6/	0 9556
C	0 9557
CCC NOTE.. THE EQUIVALENCE STATEMENT MUST READ AS FOLLOWS	0 9558
CCC ((WHEN EXTENDED, OR Q-PRECISION))	0 9559
CC EQUIVALENCE (Q(1),W1(1)), (Q(5001),W2(1))	0 9560
C	0 9561
IF (LEQU) WRITE(NOT,1000)	0 9562
1000 FORMAT (' SUBROUTINE QRRVR ')	0 9563
C PUT A INTO W2	0 9564
C	0 9565
DO 10 I=1,N	0 9566
DO 10 J=1,N	0 9567
10 W2(I,J) = A(I,J)	0 9568
C	0 9569
C NOW PLACE W2 INTO Q (AS IF FOR EXTENDED PRECISION)	0 9570
C -- --	0 9571
DO 20 J=1,N	0 9572
DO 20 I=1,N	0 9573
20 Q(I,J) = W2(I,J)	0 9574
IF (.NOT.LEQU) GO TO 40	0 9575
WRITE(NOT,1001)	0 9576
1001 FORMAT (' ELEMENTS OF ARRAY Q')	0 9577
CALL WRITES(Q,N,N,KR)	0 9578
40 CONTINUE	0 9579
C	0 9580
C PUT MATRIX INTO UPPER HESSENBERG FORM	0 9581
CALL SUBDIA (Q,N,KR,V1)	0 9582
IF (.NOT.LEQU) GO TO 50	0 9583
WRITE(NOT,1002)	0 9584
1002 FORMAT (' UPPER HESSENBERG FORM OF Q OUT OF SUBDIA IS')	0 9585
CALL WRITES(Q,N,N,KR)	0 9586
50 CONTINUE	0 9587
C	0 9588
C CALCULATE EIGENVALUES USING QR METHOD	0 9589
CALL QRCON (Q,N,KR,RI,KR,0.0)	0 9590
C	0 9591
C ALIGN EIGENVALUES INTO INCREASING ORDER	0 9592
C	0 9593
NM1 = N-1	0 9594
DO 35 J=1,NM1	0 9595
W2MIN = RR(J)	0 9596
WMIN = RI(J)	0 9597
IMIN = J	0 9598
JP1 = J+1	0 9599
DO 30 I=JP1,N	0 9600
IF (W2MIN .LE. RR(I)) GO TO 30	0 9601
W2MIN = RR(I)	0 9602
WMIN = RI(I)	0 9603
IMIN = I	0 9604
30 CONTINUE	0 9605
RR(IMIN) = RR(J)	0 9606
RI(IMIN) = RI(J)	0 9607
RI(J) = WMIN	0 9608
35 RR(J) = W2MIN	0 9609
IF (.NOT.LEQU) GO TO 60	0 9610
WRITE(NOT,1004)	0 9611

1004	FORMAT (' REAL ROOTS OUT OF QRCON ARE ')	0 9612
	CALL WRITES(RR,1,N,1)	0 9613
	WRITE(101,1005)	0 9614
1005	FORMAT (' IMAGINARY ROOTS OUT OF QRCON ARE ')	0 9615
	CALL WRITES(RI,1,N,1)	0 9616
60	CONTINUE	0 9617
	RETURN	0 9618
	END	0 9619
	SUBROUTINE QRCON (A,M,ROOTR,ROOTI,KR,IPRNT,ITIMES)	0 9620
C		0 9621
C	DEBUG # 85	0 9622
	IMPLICIT REAL*8(A-H,O-Z)	0 9623
C	PROGRAM TO CALL QR TRANSFORMATION. MAXIMUM ITER IS 50.	0 9624
	DIMENSION A(KR,1),ROOTR(1),ROOTI(1)	0 9625
	TEST=10.00**(-20)*10.00**(1*ITIMES)	0 9626
	IF (ITIMES .EQ. 0) TEST=10.00**(-20)	0 9627
	N = M	0 9628
	IF(IPRNT) 100, 110, 100	0 9629
100	WRITE (6,104)	0 9630
110	ZERO = 0.00	0 9631
	JJ=1	0 9632
120	XNN=0.00	0 9633
	XN2=0.00	0 9634
	AA = 0.00	0 9635
	E = 0.00	0 9636
	C = 0.00	0 9637
	CU = 0.00	0 9638
	R=0.00	0 9639
	SIC=0.00	0 9640
	ITER = 0	0 9641
130	IF(N-2) 140, 160, 190	0 9642
140	IF(IPRNT) 150, 160, 150	0 9643
150	WRITE (6,105)A(1,1)	0 9644
160	ROOTR(1) = A(1,1)	0 9645
	ROOTI(1) = 0.00	0 9646
170	RETURN	0 9647
180	JJ=-1	0 9648
190	X = (A(N-1,N-1) - A(N,N))**2	0 9649
	S = 4.00*A(N,N-1)*A(N-1,N)	0 9650
	ITER = ITER + 1	0 9651
	IF (X .LC. 0.00) GO TO 240	0 9652
	IF (DABS(S/X) .GT. 1.00-8) GO TO 240	0 9653
200	IF(DABS(A(N-1,N-1))-DABS(A(N,N))) 220, 220, 210	0 9654
210	E = A(N-1,N-1)	0 9655
	G = A(N,N)	0 9656
	GO TO 230	0 9657
220	G = A(N-1,N-1)	0 9658
	E = A(N,N)	0 9659
230	F = 0.00	0 9660
	H = 0.00	0 9661
	GO TO 290	0 9662
240	S = X + S	0 9663
	X = A(N-1,N-1) + A(N,N)	0 9664
	IF(S) 260, 250, 250	0 9665

250 SQ=DSQRT(S)	0 9666
F=C.C0	0 9667
H=0.C0	0 9668
IF (X) 260, 260, 270	0 9669
260 E=(X-SQ)/2.00	0 9670
G=(X+SQ)/2.00	0 9671
GO TC 250	0 9672
270 G=(X-SQ)/2.00	0 9673
E=(X+SQ)/2.00	0 9674
GO TC 290	0 9675
280 F=DSQRT(-S)/2.00	0 9676
E=X/2.00	0 9677
G=E	0 9678
H=-F	0 9679
290 IF(JJ) 310, 300, 300	0 9680
300 C = TEST *(DABS(G) + F)	0 9681
IF(DABS(A(N-1,N-2)) .GT. D) GO TO 340	0 9682
310 IF(IFRNT) 320, 330, 320	0 9683
320 WRITE (6,105)E,F, ITER	0 9684
WRITE (6,105)G,H	0 9685
330 ROCTR(N) = E	0 9686
ROCTI(N) = F	0 9687
ROCTR(N-1) = G	0 9688
ROCTI(N-1) = H	0 9689
N=N-2	0 9690
IF(JJ) 170, 120, 120	0 9691
340 IF(DABS(A(N,N-1)) .GT. TEST *DABS(A(N,N))) GO TO 380	0 9692
350 IF(IFRNT) 360, 370, 360	0 9693
360 WRITE (6,105)A(N,N), ZERO, ITER	0 9694
370 ROCTR(N) = A(N,N)	0 9695
ROCTI(N) = 0.00	0 9696
N=N-1	0 9697
GO TC 120	0 9698
380 IF(DABS(DABS(XNN/A(N,N-1))-1.00)-1.00-8) 400, 400, 390	0 9699
390 IF(DABS(DABS(XN2/A(N-1,N-2))-1.00)-1.00-8) 400, 400, 490	0 9700
400 VO=DABS(A(N,N-1))-DABS(A(N-1,N-2))	0 9701
IF (ITER-15) 520, 410, 440	0 9702
410 IF(VG) 420, 420, 430	0 9703
420 R = A(N-1,N-2)**2	0 9704
SIG = 2.00*A(N-1,N-2)	0 9705
GO TC 570	0 9706
430 R = A(N,N-1)**2	0 9707
SIG = 2.00*A(N,N-1)	0 9708
GO TC 570	0 9709
440 IF(VG) 470, 470, 450	0 9710
450 IF(IFRNT) 460, 330, 460	0 9711
460 WRITE (6,107)A(N-1,N-2)	0 9712
GO TC 320	0 9713
470 IF(IFRNT) 480, 370, 480	0 9714
480 WRITE (6,107)A(N,N-1)	0 9715
GO TC 360	0 9716
490 IF(ITER .GT. 50) GO TO 400	0 9717
IF(ITER .GT. 5) GO TO 520	0 9718
500 R=E*E+F*F	0 9719
Z1= (E-AA) *(E-AA)+(F-B) *(F-B)	0 9720
IF (R .NE. 0.00) GO TO 501	0 9721
Z1 = 0.00	0 9722
GO TC 502	0 9723
501 Z1 = Z1/R	0 9724
502 R=G*G+H*H	0 9725

Z2 = (G-C)*(G-C)+(H-D)*(H-D)	0 9726
IF (S .LE. 0.00) GO TO 503	0 9727
Z2 = C.DC	0 9728
GO TO 504	0 9729
503 Z2=Z2/R	0 9730
504 CONTINUE	0 9731
IF (Z1-J,2500) 510, 515, 540	0 9732
510 IF (Z2-J,2500) 520, 525, 530	0 9733
520 R=C*(C-F#F-	0 9734
SIG=E+J	0 9735
GO TO 570	0 9736
530 R=E*E	0 9737
SIG=E+L	0 9738
GO TO 570	0 9739
540 IF (Z2-J,2500) 550, 555, 560	0 9740
550 R=G*G	0 9741
SIG=C+G	0 9742
GO TO 570	0 9743
560 K = C.DC	0 9744
SIG = J.DC	0 9745
570 XNN=A(N,N-1)	0 9746
XN2=A(N-1,N-2)	0 9747
CALL GR2 (A,N,R,SIG,0,KR)	0 9748
AA=E	0 9749
B=F	0 9750
C=G	0 9751
D=H	0 9752
GO TO 150	0 9753
104 FORMAT(///1X, 9HREAL PART 6X 14HIMAGINARY PART, 20X	0 9754
1 13HTAKEN AS ZERO 6X 4HITER //)	0 9755
105 FORMAT(1X,015.8,3X,015.8, 42X 13)	0 9756
107 FORMAT(50X 013.8)	0 9757
END	0 9758

SUBROUTINE SUBDIA (A,M,KR,G)	0 9759
C	0 9760
IMPLICIT REAL*8(A-H,O-Z)	0 9761
DIMENSION A(KR,1),B(1)	0 9762
C SUBROUTINE TO PUT MATRIX IN UPPER HESSENBERG FORM.	0 9763
IF (M - 2) 260, 260, 100	0 9764
100 GO 250 LC = 3.M	0 9765
N = M - LC + 3	0 9766
N1 = N - 1	0 9767
N2 = N - 2	0 9768
N1 = N1	0 9769
DIV = DABS(A(N,N-1))	0 9770
GO 120 J = 1,N2	0 9771
IF(DABS(A(N,J))- DIV) 120, 120, 110	0 9772
110 N1 = J	0 9773
DIV = DABS(A(N,J))	0 9774
120 CONTINUE	0 9775
IF(DIV) 130, 250, 130	0 9776
130 IF(N1 - N1) 140, 170, 140	0 9777
140 GO 150 J = 1,N	0 9778
DIV = A(J,N1)	0 9779

A(J,N1) = A(J,N1)	0 9780
150 A(J,N1) = DIV	0 9781
DO 160 J = 1,M	0 9782
DIV = A(N1,J)	0 9783
A(N1,J) = A(N1,J)	0 9784
160 A(N1,J) = DIV	0 9785
170 DO 180 K = 1, N1	0 9786
180 E(K) = A(N,K)/A(N,N-1)	0 9787
DO 240 J = 1,M	0 9788
SUM = J.D0	0 9789
IF (J - N1) 190, 220, 220	0 9790
190 IF(B(J)) 200, 220, 230	0 9791
200 A(N,J) = 0.D0	0 9792
DO 210 K = 1,N1	0 9793
A(K,J) = A(K,J) - A(K,N1)*B(J)	0 9794
210 SUM = SUM + A(K,J)*B(K)	0 9795
GO TC 240	0 9796
220 DO 230 K = 1,N1	0 9797
230 SUM = SUM + A(K,J)*B(K)	0 9798
240 A(N1,J) = SUM	0 9799
250 CONTINUE	0 9800
260 RETURN	0 9801
END	0 9802

SUBROUTINE QR2 (A,N,R,SIG,D,KR)	0 9803
C	0 9804
SUBROUTINE TO BE USED BY QRCON	0 9805
IMPLICIT REAL*8(A-H,O-Z)	0 9806
DIMENSION A(KR,1),G(3),PSI(2)	0 9807
N1 = N - 1	0 9808
IA = N - 2	0 9809
IP = IA	0 9810
IF(N-3) 450, 140, 100	0 9811
100 DO 130 J = 3,N1	0 9812
J1 = N - J	0 9813
IF(DABS(A(J1+1,J1))-D) 140, 140, 110	0 9814
110 DEN = A(J1+1,J1+1)*(A(J1+1,J1+1)-SIG)+A(J1+1,J1+2)*A(J1+2,J1+1)+R	0 9815
IF(DEN) 120, 130, 120	0 9816
120 IF(DABS(A(J1+1,J1)*A(J1+2,J1+1)*(DABS(A(J1+1,J1+1)+A(J1+2,J1+2)	0 9817
1-SIG)+DABS(A(J1+3,J1+2)))/DEN)-D) 140, 140, 130	0 9818
130 IP=J1	0 9819
140 DO 150 J=1,IP	0 9820
J1=IP-J+1	0 9821
IF (DABS(A(J1+1,J1))-D) 160, 160, 150	0 9822
150 IQ=J1	0 9823
160 DO 440 I=IP,N1	0 9824
IF(I-IP) 180, 170, 180	0 9825
170 G(1)=A(IP,IP)*(A(IP,IP)-SIG)+A(IP,IP+1)*A(IP+1,IP)+R	0 9826
G(2)=A(IP+1,IP)*(A(IP,IP)+A(IP+1,IP+1)-SIG)	0 9827
G(3)=A(IP+1,IP)*A(IP+2,IP+1)	0 9828
A(IP+2,IP)=0.D0	0 9829
GO TC 210	0 9830
180 G(1)=A(I,I-1)	0 9831
G(2)=A(I+1,I-1)	0 9832
IF(I-IA) 190, 190, 200	0 9833

190	G(3)=A(I+2,I-1)	0 9834
	GO TC 210	0 9835
200	G(3)=0.00	0 9836
210	XK=DSORT(G(1)*G(1)+G(2)*G(2)+G(3)*G(3))	0 9837
	IF(G(1).LT.0.0D0) XK=-XK	0 9838
220	IF(XK) 230, 240, 230	0 9839
230	AL=G(1)/XK+1.00	0 9840
	PSI(1)=G(2)/(G(1)+XK)	0 9841
	PSI(2)=G(3)/(G(1)+XK)	0 9842
	GO TC 250	0 9843
240	AL=2.00	0 9844
	PSI(1)=0.00	0 9845
	PSI(2)=0.00	0 9846
250	IF(I-IQ) 260, 290, 260	0 9847
260	IF(I-IP) 260, 270, 280	0 9848
270	A(I,I-1)=-A(I,I-1)	0 9849
	GO TC 290	0 9850
280	A(I,I-1)=-XK	0 9851
290	DO 340 J=1,N	0 9852
	IF(I-1A) 300, 300, 310	0 9853
300	C=PSI(2)*A(I+2,J)	0 9854
	GO TC 320	0 9855
310	C=0.00	0 9856
320	E=AL*(A(I,J)+PSI(1)*A(I+1,J)+C)	0 9857
	A(I,J)=A(I,J)-E	0 9858
	A(I+1,J)=A(I+1,J)-PSI(1)*E	0 9859
	IF(I-1A) 330, 330, 340	0 9860
330	A(I+2,J)=A(I+2,J)-PSI(2)*E	0 9861
340	CONTINUE	0 9862
	IF(I-1A) 350, 350, 360	0 9863
350	L=I+2	0 9864
	GO TC 370	0 9865
360	L=N	0 9866
370	DO 420 J=IQ,L	0 9867
	IF(I-1A) 380, 380, 390	0 9868
380	C=PSI(2)*A(J,I+2)	0 9869
	GO TC 400	0 9870
390	C=0.00	0 9871
400	E=AL*(A(J,I)+PSI(1)*A(J,I+1)+C)	0 9872
	A(J,I)=A(J,I)-E	0 9873
	A(J,I+1)=A(J,I+1)-PSI(1)*E	0 9874
	IF(I-1A) 410, 410, 420	0 9875
410	A(J,I+2)=A(J,I+2)-PSI(2)*E	0 9876
420	CONTINUE	0 9877
	IF(I-N+J) 430, 430, 440	0 9878
430	E=AL*PSI(2)*A(I+3,I+2)	0 9879
	A(I+3,I)=-E	0 9880
	A(I+3,I+1)=-PSI(1)*E	0 9881
	A(I+3,I+2)=A(I+3,I+2)-PSI(2)*E	0 9882
440	CONTINUE	0 9883
450	RETURN	0 9884
	END	0 9885

SUBROUTINE RWRITE (K,KR1,RI1,PR2,RI2,N1,N2,ANAM1,ANAM2)

0 9886
0 9887

C

C		DEBUG # 88	0 9888
	IMPLICIT REAL*8(A-H,O-Z)		0 9889
C			0 9890
C	---LATEST MOD ON 09/04/75		0 9891
C	---LATEST MOD ON 09/04/75		0 9892
C	-----SUBROUTINE PULLS UP NEW PAGE VIA PAGEHD, PRINTS OUT		0 9893
C	IDENTIFICATION(S), ANAM1 AND ANAM2, THEN PRINTS ROOTS.		0 9894
C			0 9895
C	-----SUBROUTINE ARGUMENT DESCRIPTIONS-----		0 9896
C			0 9897
C	ALL ARGUMENTS ARE INPUT		0 9898
C			0 9899
C	K = NO. OF ROOT SETS TO PRINT.		0 9900
C	RR1 = REAL ROOTS (FIRST SET)		0 9901
C	RI1 = IMAG ROOTS (FIRST SET)		0 9902
C	RR2 = REAL ROOTS (SECOND SET)		0 9903
C	RI2 = IMAG ROOTS (SECOND SET)		0 9904
C	N1 = ROOT COUNT (FIRST SET)		0 9905
C	N2 = ROOT COUNT (SECOND SET)		0 9906
C			0 9907
C	ANAM1 = 4 CHARACTER ALPHANUMERIC TITLE (FIRST SET)		0 9908
C	ANAM2 = 4 CHARACTER ALPHANUMERIC TITLE (SECOND SET)		0 9909
C			0 9910
	DIMENSION RR1(1), RI1(1), RR2(1), RI2(1)		0 9911
	DATA NCT /		0 9912
	1 6 /		0 9913
C			0 9914
	101 FORMAT (///20X,A4,34X,A4,///5X,2HNO,5X,9HREAL PART ,		0 9915
	1 3X,14HIMAGINARY PART,11X,9HREAL PART,3X,14HIMAGINARY PART,		0 9916
	2 //)		0 9917
	102 FORMAT (5X,12,3X,D12.5,5X,D12.5,9X,D12.5,3X,D12.5)		0 9918
	103 FORMAT (5X,12,41X,D12.5,5X,D12.5)		0 9919
C			0 9920
	CALL PAGEHD		0 9921
	IF (K .EQ. 2) GO TO 20		0 9922
	WRITE (NCT,101) ANAM1		0 9923
	IL = 0		0 9924
	DO 10 I=1,N1		0 9925
	IL = IL+1		0 9926
	IF (IL .LE. 45) GO TO 10		0 9927
	IL = 0		0 9928
	CALL PAGEHD		0 9929
	WRITE (NCT,101) ANAM1		0 9930
	10 WRITE (NCT,102) I,RR1(I), RI1(I)		0 9931
	RETURN		0 9932
	20 CONTINUE		0 9933
C			0 9934
	L = MAX0(N1,N2)		0 9935
	WRITE (NCT,101) ANAM1, ANAM2		0 9936
	IL = 0		0 9937
	DO 40 I=1,L		0 9938
	IL=IL+1		0 9939
	IF (IL .LE. 45) GO TO 25		0 9940
	IL=0		0 9941
	CALL PAGEHD		0 9942
	WRITE (NCT,101) ANAM1, ANAM2		0 9943
	25 CONTINUE		0 9944
	IF (I .GT. N1 .OR. I .GT. N2) GO TO 30		0 9945
	WRITE (NCT,102) I, RR1(I),RI1(I),RR2(I),RI2(I)		0 9946
	GO TO 40		0 9947

```

30 CONTINUE                                0 9948
    IF (I.GT. N2) WRITE (NUT,102) I, RR1(I), R11(I) 0 9949
    IF (I.GT. N1) WRITE (NUT,103) I, RR2(I), R12(I) 0 9950
40 CONTINUE                                0 9951
C                                           0 9952
    RETURN                                    0 9953
    END                                      0 9954

SUBROUTINE SPLOT (TITLE,FMAX,FMIN,DBMIN,DBMAX) 0 9955
C                                           DEBUG # 89 0 9956
C                                           0 9957
    REAL*8 IRUNNO, IDATE                    0 9958
    REAL*8 UNAME, TITLE1, TITLE2           0 9959
C SUBROUTINE PLOTS FREQUENCY DATA ON SEMILOG GRID. 0 9960
C                                           0 9961
C -----SUBROUTINE ARGUMENT DESCRIPTIONS----- 0 9962
C TITLE = INPUT 80 CHARACTER ALPHANUMERIC TITLE. 0 9963
C FMAX = INPUT UPPER RANGE LIMIT FOR FREQUENCY SWEEP. 0 9964
C FMIN = INPUT LOWER RANGE LIMIT FOR FREQUENCY SWEEP. 0 9965
C DBMIN = MINIMUM DB TO PLOT. 0 9966
C DBMAX = MAXIMUM DB TO PLOT. 0 9967
C                                           0 9968
    DIMENSION TITLE(1),DBTITLE(36),FTITLE(7),ANUM(9) 0 9969
    DIMENSION ZXI(2), ZYI(2) 0 9970
    DIMENSION FAZL(500),DBAMP(500) 474 9971
    COMMON /LSTART/ IRUNNO, IDATE, NPAGE 0 9972
    COMMON /LSTRT1/ UNAME(5), TITLE1(12), TITLE2(12) 0 9973
    COMMON /LZMODE/ AMODE(200) 0 9974
    COMMON /RSTUFF/ 0 9975
    * SAVED(500), SAVEP(500), SAVED(500), SAVEA(500), KSAVE 447 9976
C                                           0 9977
    DATA DBTITL / 0 9978
    * 4F-200,4H-190,4H-180,4H-170,4H-160,4H-150,4H-140,4H-130,4H-120, 0 9979
    * 4F-110,4H-100,4H-90,4H-80,4H-70,4H-60,4H-50,4H-40,4H-30, 0 9980
    * 4F-20,4H-10,4H 0.0,4H 10,4H 20,4H 30,4H 40,4H 50,4H 60, 0 9981
    * 4F 70,4H 80,4H 90,4H100,4H110,4H120,4H130,4H140,4H150./ 0 9982
C                                           0 9983
    DATA FTITLE / 0 9984
    * 4HE-2 ,4HE-1 ,4HE00 ,4HE01 ,4HE02 ,4HE03 ,4HE04 / 0 9985
C                                           0 9986
    DATA ANUM / 0 9987
    * 1H1,1H2,1H3,1H4,1H5,1H6,1H7,1H8,1H9 / 0 9988
C                                           0 9989
    CALL RSETMG (AMODE) 0 9990
C                                           0 9991
C-----ESTABLISH INITIAL DECADE FACTOR 0 9992
    AG = ALOG10(FMIN) 0 9993
    IF ((AG.LT.0.) .AND. (AG-FLCAT(IFIX(AG)).NE. J.)) AG = AG-1. 0 9994
    FACTOR = 10.**IFIX(AG) 0 9995
    FMIN = FACTOR 0 9996
C                                           0 9997
C DETERMINE GRID FREQUENCY 0 9998
    AG = ALOG10(FMAX) 0 9999
    IF(AG.GE. 0.0 .AND. (AG-FLCAT(IFIX(AG))) .NE. 0.0) 010000
    I AG = AG + 1.0 010001

```


K = ALOG10(FMIN)	010002
K = K-1	010003
C	010004
UP = 10. **IFIX(AG)	010005
NLC = IFIX(AG) - IFIX(ALOG10(FMIN))	010006
C	010007
CALL SCALE (DBMAX,DBMIN,10,DBU,DBL,ZIP)	010008
IF (DBL .LT. -200.) DBL = -200.	010009
C	010010
C--CALCULATE LABEL LOCATORS.	010011
C	010012
ISY = (DBL + 200.)/10. + 1.	010013
ISX = ALOG10(FMIN) + 3.0	010014
C	010015
C	010016
C MODIFY PHASE VALUES TO PLOT USING DB MAPPING SCALE.	010017
R1 = (DBL-DBL)* 0.0025	010018
R2 = (DBU+DBL)* 0.5	010019
C	010020
DO 1 I=1,KSAVE	010021
YDB = SAVED(I)	010022
IF (YDB .LT. DBL) YDB = DBL	010023
IF (YDB .GT. DBU) YDB = DBU	010024
DBAMP(I) = YDB	010025
XFAZE = SAVED(I)	010026
IF (XFAZE .GT. 180.) XFAZE = XFAZE - 360.	010027
1 FAZE(I) = R1*XFAZE + R2	010028
C	010029
C	010030
C GENERATE PLOT GRID WITH SCALES, LABELS, AND TITLE.	010031
C	010032
C INDICATE UTILIZATION OF NORMALIZED SPACE	010033
C	010034
CALL SETSMG (AMODE,19,1.)	010035
CALL SETSMG (AMODE,20,1.)	010036
C	010037
C	010038
C USE FULL SPACES FOR PUTTING ON RIGHT HAND LABELS.	010039
C	010040
CALL OBJECTG (AMODE,0.,0.,1.,1.)	010041
CALL SUBJEG (AMODE,0.,0.,1.,1.)	010042
C--BOX THE GRID.	010043
CALL BOXG (AMODE,0.0,0.0,0.0,1.0,0.0,0.0,0.99,1.0,0.99)	010044
C--INCREASE CHARACTER SIZE	010045
CALL SETSMG (AMODE,45,1.5)	010046
C--PLT ON THE BOXE TITLE.	010047
CALL LEGNDG (AMODE,0.45,0.93,4,4HBCDE)	010048
C	010049
C--SET FLAG TO ORIENT LABEL AT 90. DEGREES.	010050
C	010051
CALL SETSMG (AMODE,46,90.)	010052
C	010053
C--PLT LABEL ON RHS.	010054
C	010055
CALL LEGNDG (AMODE,0.960,0.4355,11,11HPHASE - DEG)	010056
C--LABEL THE Y-AXIS.	010057
CALL LEGNDG (AMODE,0.06,0.44,9,9HGAIN - DB)	010058
C	010059
C--RESET CHARACTER SIZE	010060
CALL SETSMG (AMODE,45,1.0)	010061

C		C10062
C--RESET ORIENTATION OF LABEL.		C10063
CALL SETSMG (AMODE,46,0.)		C10064
C		C10065
C--PLACE SCALE ON RHS. (USING LEGNDG)		C10066
C		C10067
CALL LEGNDG (AMODE,.907,.910,4,4H 200)		C10068
CALL LEGNDG (AMODE,.907,.832,4,4H 160)		C10069
CALL LEGNDG (AMODE,.907,.754,4,4H 120)		C10070
CALL LEGNDG (AMODE,.907,.676,4,4H 80)		C10071
CALL LEGNDG (AMODE,.907,.598,4,4H 40)		C10072
CALL LEGNDG (AMODE,.907,.520,4,4H 0)		C10073
CALL LEGNDG (AMODE,.907,.442,4,4H -40)		C10074
CALL LEGNDG (AMODE,.907,.364,4,4H -80)		C10075
CALL LEGNDG (AMODE,.907,.286,4,4H -120)		C10076
CALL LEGNDG (AMODE,.907,.208,4,4H -160)		C10077
CALL LEGNDG (AMODE,.907,.130,4,4H -200)		C10078
C		C10079
C--PLT ON ADDITIONAL TITLES.		C10080
CALL SETSMG (AMODE,45,1,25)		C10081
CALL LINESG (AMODE,0,0,05,0,02)		C10082
CALL TEXTG (AMODE,8,1RUNNO)		C10083
CALL TEXTG (AMODE,6,1DATE)		C10084
CALL LINESG (AMODE,0,0,0,0,02)		C10085
CALL TEXTG (AMODE,0,UNAME(1))		C10086
CALL TEXTG (AMODE,0,UNAME(2))		C10087
CALL TEXTG (AMODE,0,UNAME(3))		C10088
CALL SETSMG (AMODE,45,1,.)		C10089
C		C10090
C ESTABLISH SUBJECT AND OBJECT SPACES.		C10091
C		C10092
CALL OBJECTG (AMODE,0,11,0,13,0,89,0,91)		C10093
CALL SUBJEG (AMODE,FMIN,DBL,UP,DBU)		C10094
C		C10095
C INDICATE NONLINEAR X-SCALE.		C10096
C		C10097
CALL SETSMG (AMODE,23,1,.)		C10098
C		C10099
C INDICATE DESIRED EMPHISIS ON Y=0 ONLY.		C10100
C		C10101
CALL SETSMG (AMODE,100,2,.)		C10102
C		C10103
C--PLT ON GRID.		C10104
C		C10105
CALL GRIDG (AMODE,1,5,0,1,2)		C10106
C		C10107
C--PLT ON TITLE AND LABELS		C10108
C		C10109
CALL SETSMG (AMODE,45,1,5)		C10110
CALL TITLG (AMODE,19,19HFREQUENCY. - RAD/SEC,		C10111
1 0,DUMY,		C10112
2 00, TITLE)		C10113
CALL SETSMG (AMODE,45,1,.)		C10114
C		C10115
C--LABEL Y-AXIS.		C10116
C		C10117
CALL LABELG (AMODE,1,10,.4,CBTITLE(1SY))		C10118
C		C10119
C--LABEL X-AXIS)		C10120
C		C10121

CALL LABELG (AMODE,0, 1.,4,FTITLE(1X))	010122
C	010123
C--SET FLAG TO ORIENT LABEL AT 90 DEGREES.	010124
C	010125
CALL SETSMG (AMODE,46,90.)	010126
C	010127
C--PUT ON LOG GRID AND INTERMEDIATE LABELS.	010128
C	010129
ZYI(1) = DBL	010130
ZYI(2) = DBU	010131
DO 7 J=1,NLC	010132
K = K+1	010133
DO 7 I=2,9	010134
XI = FLOAT(1) * 10. **K	010135
ZXI(1) = XI	010136
ZXI(2) = XI	010137
CALL LINESS (AMODE,2,ZXI,ZYI)	010138
C	010139
CALL OBJCTG (AMODE,0,11,0.,0.89,1.)	010140
CALL SUBJEG (AMODE,FMIN,0.,UP,1.)	010141
CALL LEGNDG (AMODE,XI,0.12,1,ANUM(1))	010142
CALL OBJCTG (AMODE,0,11,0.13,0.89,0.91)	010143
CALL SUBJEG (AMODE,FMIN,DBL,UP,DBU)	010144
C	010145
7 CONTINUE	010146
C	010147
C--RESET ORIENTATION OF LABEL.	010148
C	010149
CALL SETSMG (AMODE,46,0.)	010150
C	010151
C--PUT DATA ON PLOT.	010152
C	010153
CALL LINESS (AMODE,KSAVE,SAVEQ,FAZE)	010154
CALL TEXTG (AMODE,1,1HP)	010155
CALL LINESS (AMODE,KSAVE,SAVEQ,DBAMP)	010156
CALL TLXTG (AMODE,1,1HG)	010157
CALL PAGEG (AMODE,0,0,1)	010158
RETURN	010159
END	010160
SUBROUTINE NIPLOT (TITLE,DBMIN,DBMAX)	010161
C	010162
REAL*8 IRUNNO, IDATE	010163
REAL*8 UNAME, TITLE1, TITLE2	010164
C	010165
C-----SUBROUTINE DISPLAYS S-PLANE FREQUENCY	010166
C-----VIA THE NICHOLS PLOT	010167
C	010168
C-----SUBROUTINE ARGUMENT DESCRIPTIONS-----	010169
C	010170
C-----TITLE = INPUT 30 CHARACTER ALPHANUMERIC TITLE.	010171
C-----DBMIN = INPUT LOWER DBLIMIT.	010172
C-----DBMAX = UPPER DBLIMIT.	010173
C	010174
DIMENSION TITLE(1), DBITL(36)	010175

20 DBL = AMIN1 (DBL,SAVED(1))	010236
IF (DBL .LT. DBMIN) DBL = DBMIN	010237
DBU = DBL + 100.	010238
C	010239
ISY = (DBL + 200.)/10. + 1.	010240
CALL SUBJEG (AMODE,-140.,DBL ,260.,DBU)	010241
C	010242
CALL GRIDG (AMODE,10.,5.,4,2)	010243
C	010244
C-----NOW HAVE CARTESIAN GRID.	010245
C-----PUT ON TITLE AND LABELS.	010246
C	010247
C	010248
CALL LABELG (AMODE,0,40.,3,	010249
1 33H240280320 0 40 80120160200240250)	010250
C	010251
CALL LABELG (AMODE,1,10.,4,DBTITL(ISY))	010252
C	010253
KCPH = 0	010254
PDPHI = 0.0	010255
C-----NOW HAVE GRID,TITLE, AND LABELS.	010256
C	010257
GO 1480 I=1,KSAVE	010258
C	010259
DPHI = SAVED(1)	010260
DBELL = SAVED(1)	010261
C	010262
IF (I .GT. 1) GO TO 1400	010263
C	010264
C-----FIRST POINT	010265
C	010266
IF (DPHI .GT. 225.) DPHI = DPHI - 360.	010267
IF (DBELL .LT. DBL .OR. DBELL .GT. DBU) GO TO 1425	010268
C	010269
C-----POSITION BEAM (AND WRITE FREQUENCY ON GRAPH)	010270
C	010271
CALL LINEG (AMODE,0,DPHI,DBELL)	010272
GO TO 1480	010273
C	010274
1400 IF (DPHI .GT. 225.) DPHI = DPHI - 360.	010275
1420 IF (DBELL .GT. DBL .AND. DBELL .LT. DBU) GO TO 1430	010276
IF (KCPH .NE. 0) GO TO 1480	010277
1425 KCPH = 1	010278
GO TO 1480	010279
1430 IF (KCPH .EQ. 0) GO TO 1440	010280
KCPH = 0	010281
GO TO 1450	010282
1440 IF (ABS(PDPHI-DPHI) .LT. 330.) GO TO 1470	010283
1450 CALL LINEG (AMODE,0,DPHI,DBELL)	010284
GO TO 1480	010285
1470 CALL LINEG (AMODE,1,DPHI,DBELL)	010286
1480 PDPHI = DPHI	010287
CALL RSETMG (AMODE)	010288
CALL PAGEG (AMODE,0,0,1)	010289
C	010290
RETURN	010291
END	010292

SUBROUTINE NYPL0T (TITLE,AMIN,AMAX)	010293
C	010294
REAL*8 IRUNNO, IDATE	010295
REAL*8 UNAME, TITLE1, TITLE2	010296
C	010297
C-----SUBROUTINE PLOTS AR VS. DPHI ON A POLAR GRID.	010298
C	010299
C	010300
C	010301
C	010302
C	010303
C	010304
C	010305
COMMON /LSTART/ IRUNNO, IDATE, NPAGE	010306
COMMON /LSTRT1/ UNAME(3), TITLE1(12), TITLE2(12)	010307
COMMON /PSTUFF/	010308
* SAVED(500), SAVEP(500), SAVED(500), SAVEA(500), KSAVE	44710309
COMMON /LZMODE/ AMODE(200)	010310
C	010311
CALL RSETMG (AMODE)	010312
C-----USE FULL SPACE FOR BOX AND TITLING.	010313
CALL SETSMG (AMODE,19,1.)	010314
CALL SETSMG (AMODE,20,1.)	010315
CALL OBJCTG (AMODE,0.,0.,1.,1.)	010316
CALL SUBJEG (AMODE,0.,0.,1.,1.)	010317
C-----BOX THE GRID.	010318
CALL BOXG (AMODE,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.99,1.0,0.99)	010319
C-----INCREASE CHARACTER SIZE.	010320
CALL SETSMG (AMODE,45,1.5)	010321
C-----PUT ON NYQUIST TITLE.	010322
CALL LEGNDG (AMODE,0.42,0.90,13,13HNYQUIST (523))	010323
C-----PUT ON TITLE.	010324
C	010325
CALL LEGNDG (AMODE,0.2,0.04,60,TITLE)	010326
C-----TITLE X-AXIS.	010327
C	010328
CALL LEGNDG (AMODE,0.4,0.08,15,15HPHASE (DEGREES))	010329
C-----TITLE Y-AXIS.	010330
C	010331
CALL SETSMG (AMODE,40,90.)	010332
CALL LEGNDG (AMODE,0.00,0.4,9,9HAMPLITUDE)	010333
CALL SETSMG (AMODE,40,0.0)	010334
C	010335
C-----PUT ON ADDITIONAL TITLING.	010336
CALL SETSMG (AMODE,45,1.25)	010337
CALL LINESG (AMODE,0.0,0.05,0.02)	010338
CALL TEXTG (AMODE,8,IRUNNO)	010339
CALL TEXTG (AMODE,6,IDATE)	010340
CALL LINESG (AMODE,0.0,0.3,0.02)	010341
CALL TEXTG (AMODE,8,UNAME(1))	010342
CALL TEXTG (AMODE,8,UNAME(2))	010343
CALL TEXTG (AMODE,8,UNAME(3))	010344
CALL SETSMG (AMODE,45,1.)	010345
CALL RSETMG (AMODE)	010346
C	010347
C-----ESTABLISH LIMITS.	010348
CALL PLTSS (AMAX,AMIN,KMAX,KMIN)	010349
IF (KMIN .LT. 0.) KMIN = 0.	010350
C	010351
C	010352

IF (FMAX .EQ. RMIN) RMAX = RMIN+10.	010353
C	010354
TMAX = 360.	010355
TMIN = 0.0	010356
C	010357
C-----SET OBJECT SPACE.	010358
CALL SETSMG (AMODE,19,1.33333)	010359
CALL SETSMG (AMODE,20,1.)	010360
CALL OBJCTG (AMODE,0.3,0.25,1.0,0.95)	010361
CALL PSUEJG (AMODE,RMIN,TMIN,RMAX,TMAX)	010362
C	010363
C-----PUT ON GRID LINES.	010364
C	010365
CALL SETSMG (AMODE,14,0.00)	010366
CALL SETSMG (AMODE,45,1.00)	010367
CALL PSETG (AMODE,0,00,01,IDTH,IITH,DLD,DLI,FMTD,FMTI)	010368
C	010369
DO 16 I=106,109	010370
16 AMODE(I) = 0.0	010371
C	010372
CALL PGRIDG (AMODE,00,01,IDTH,IITH)	010373
CALL PLABELG (AMODE,0,0LD,0,FMTD)	010374
CALL PLABELG (AMODE,1,DLI,0,FMTI)	010375
C	010376
DO 80 I=1,NSAVE	010377
R = SAVEA(I)	010378
T = SAVEP(I)	010379
IF (I .GT. 1) GO TO 19	010380
C	010381
C-----FIRST POINT	010382
C	010383
IF (R .LT. RMIN .OR. R .GT. RMAX) GO TO 29	010384
C	010385
C-----POSITION BEAM	010386
CALL PLINEG (AMODE,0,R,T)	010387
GO TO 30	010388
19 CONTINUE	010389
20 IF (R .GT. RMIN .AND. R .LT. RMAX) GO TO 30	010390
IF (KOPH .NE. 0) GO TO 60	010391
25 KOPH = 1	010392
GO TO 80	010393
30 IF (KOPH .EQ. 0) GO TO 70	010394
KOPH = 0	010395
GO TO 50	010396
50 CALL PLINEG (AMODE,0,R,T)	010397
GO TO 30	010398
70 CALL PLINEG (AMODE,1,R,T)	010399
80 CONTINUE	010400
C	010401
CALL PAGES (AMODE,0,1,1)	010402
CALL RSETMG (AMODE)	010403
C	010404
RETURN	010405
END	010406

```

SUBROUTINE RLPLUT (TITLE,ISNIM,ICYC,IRLC)                                010407
C                                                                           010408
C      REAL*8 IRUNNO, IDATE                                           010409
C      REAL*8 UNAME, TITLE1, TITLE2                                    010410
C                                                                           010411
C-----SUBROUTINE PLUTS ROOT LOC1 FOR A SINGLE ROOT.                  010412
C                                                                           010413
C      -----SUBROUTINE ARGUMENT DESCRIPTIONS-----                  010414
C                                                                           010415
C      TITLE = INPUT ALPHA NUMERIC TITLE                               010416
C      ISNIM = INPUT ROOT IDENTIFICATION PARAMETER.                    010417
C      =1    STARTING POINT IS OPEN LOOP ZERO.                        010418
C      =2    STARTING POINT IS OPEN LOOP POLE.                        010419
C      ICYC = INPUT TRANSFER FUNCTION COUNT (PLACED ON PLUT).          010420
C      IRLC = INPUT ROOT LOCUS CYCLE NUMBER (PUT ON PLUT)              010421
C                                                                           010422
C                                                                           010423
C      DIMENSION TITLE(1), X(500), Y(500)                             47210424
C      DIMENSION IGR (99)                                              010425
C      COMMON /LZMODE/ AMODE(200)                                     010426
C      COMMON /PSTUFF/                                                010427
C      *      SAVED(500), SAVEP(500), SAVED(500), SAVEA(500), KSAVE 44710428
C      COMMON /LSTART/ IRUNNO, IDATE, NPAGE                            010429
C      COMMON /LSTRT1/ UNAME(3), TITLE1(12), TITLE2(12)              010430
C                                                                           010431
C      EQUIVALENCE (SAVEA(1),X(1)), (SAVEP(1),Y(1))                  010432
C      DATA IGR /                                                    010433
1      2H 1,2H 2,2H 3,2H 4,2H 5,2H 6,2H 7,2H 8,2H 9,2H10,          010434
2      2H11,2H12,2H13,2H14,2H15,2H16,2H17,2H18,2H19,2H20,          010435
3      2H21,2H22,2H23,2H24,2H25,2H26,2H27,2H28,2H29,2H30,          010436
4      2H31,2H32,2H33,2H34,2H35,2H36,2H37,2H38,2H39,2H40,          010437
5      2H41,2H42,2H43,2H44,2H45,2H46,2H47,2H48,2H49,2H50,          010438
6      2H51,2H52,2H53,2H54,2H55,2H56,2H57,2H58,2H59,2H60,          010439
7      2H61,2H62,2H63,2H64,2H65,2H66,2H67,2H68,2H69,2H70,          010440
8      2H71,2H72,2H73,2H74,2H75,2H76,2H77,2H78,2H79,2H80,          010441
9      2H81,2H82,2H83,2H84,2H85,2H86,2H87,2H88,2H89,2H90,          010442
A      2H91,2H92,2H93,2H94,2H95,2H96,2H97,2H98,2H99 /              010443
C                                                                           010444
C      KSAVE = KSAVE - 1                                              010445
C                                                                           010446
C      CALL RSETMG (AMODE)                                             010447
C-----ECX (RID AND PUT ON EXTRA TITLING USING FULL SPACE.          010448
C      CALL SETSMG (AMODE,19,1.)                                       010449
C      CALL SETSMG (AMODE,20,1.)                                       010450
C      CALL OBJECTG (AMODE,0.,0.,1.,1.)                                010451
C      CALL SUBJES (AMODE,0.,0.,1.,1.)                                010452
C      CALL BOXG (AMODE,0,0,0,0,1,0,0,0,99,1,0,99)                   010453
C                                                                           010454
C-----INCREASE CHARACTER SIZE                                       010455
C      CALL SETSMG (AMODE,45,1.5)                                       010456
C      CALL LEGNUG (AMODE,0.45,0.94,10,10-ROOT LOCUS)               010457
C                                                                           010458
C-----ADDITIONAL TITLING                                           010459
C      CALL SETSMG (AMODE,45,1.25)                                       010460
C      CALL LINESG (AMODE,0,0,05,0,02)                                   010461
C      CALL TEXTG (AMODE,6,IRUNNO)                                       010462
C      CALL TEXTG (AMODE,6,IDATE)                                         010463
C      CALL LINESG (AMODE,0,0,3,0,02)                                   010464
C      CALL TEXTG (AMODE,6,UNAME(1))                                       010465
C      CALL FLXTG (AMODE,6,UNAME(2))                                       010466

```


CALL TEXTG (AMODE,6,UNAME(3))	010467
CALL LINESG (AMODE,0,0.8,0.06)	010468
CALL TEXTG (AMODE,7,7HICYC =)	010469
CALL TEXTG (AMODE,2,ISR(ICYC))	010470
CALL LINESG (AMODE,0,0.8,0.02)	010471
CALL TEXTG (AMODE,7,7HIRLC =)	010472
CALL TEXTG (AMODE,2,ISR(IRLC))	010473
C	010474
C-----TITLE IMAG AXIS ON RHS.	010475
C ORIENTATE LABEL AT 90 DEGREES.	010476
CALL SETSMG (AMODE,46,90.)	010477
CALL LEGNDG (AMODE,0.96,0.42,14,14HIMAGINARY PART)	010478
CALL SETSMG (AMODE,46,0.)	010479
C	010480
C ESTABLISH LIMITS ON DATA.	010481
C	010482
XMN = X(1)	010483
XMAX = XMN	010484
YMN = Y(1)	010485
YMAX = YMN	010486
C	010487
DO 10 I=2,KSAVE	010488
XMN = AMIN1(XMN,X(I))	010489
XMAX = AMAX1(XMAX,X(I))	010490
YMN = AMIN1(YMN,Y(I))	010491
10 YMAX = AMAX1(YMAX,Y(I))	010492
C	010493
C SET UP SUBJECT SPACE.	010494
CALL PLOTSS (XMAX,XMN,XTOP,XBOT)	010495
CALL PLOTSS (YMAX,YMN,YTOP,YBOT)	010496
CALL SUBJEG (AMODE,XBOT,YBOT,XTOP,YTOP)	010497
C	010498
C--COMPUTE ARGUMENTS TO USE IN GRIDG AND LABELG.	010499
C	010500
CALL SUTPG (AMODE,1,DX,DY,IXTH,JYTH,DLX,DLY,XFMT,YFMT)	010501
C	010502
C--RESET GRID MARGINS TO 0.	010503
C	010504
DO 60 I=106,109	010505
60 AMODE(I) = 0.	010506
C	010507
C--DRAW THE GRID.	010508
C	010509
CALL GRIDG (AMODE,DX,DY,IXTH,JYTH)	010510
C	010511
C--NOW LABEL IT.	010512
C	010513
CALL LABELG (AMODE,0,DLX,0,XFMT)	010514
CALL LABELG (AMODE,1,DLY,0,YFMT)	010515
C	010516
C--NEXT, TITLE IT.	010517
C	010518
CALL TITLG (AMODE,9,9HREAL PART,	010519
1 0,0UMTLE,	010520
2 0,0,TITLE)	010521
C	010522
C-----USE TEXTG TO INDICATE STARTING POINT.	010523
CALL LINESG (AMODE,0,X(1),Y(1))	010524
IF (ISNIM .EQ. 1) CALL TEXTG (AMODE,1,1H0)	010525
IF (ISNIM .EQ. 2) CALL TEXTG (AMODE,1,1HX)	010526

C		010527
C-----PLOT THE LUCI.		010528
C		010529
CALL LINESG (AMODE,KSAVL,X,Y)		010530
CALL LINESG (AMODE,KSAVE,X,Y)		010531
C		010532
C-----PAGE THE FRAME.		010533
C		010534
CALL PAGES (AMODE,C,O,I)		010535
RETURN		010536
END		010537

	SUBROUTINE SCALE(AMAX,AMIN,NSQ,AU,AL,S)	010538
C		010539
C		010540
C	AMAX=MAX VALUE TO BE PLOTTED	010541
C	AMIN=MIN VALUE TO BE PLOTTED	010542
C	NSQ=NUMBER OF MAJOR GRIDS = 10 GENERALLY	010543
C	AL=COMPUTED UPPER SCALE LIMIT	010544
C	AL=COMPUTED LOWER SCALE LIMIT	010545
C	S=SCALE FACTOR PER MAJOR GRID	010546
C		010547
	FNSQ=NSQ	010548
	AL=0.	010549
	IF (AMAX.GT.AMIN) GO TO 13	010550
	AMIN=AMIN-S.	010551
	AMAX=AMAX+S.	010552
13	ADIF=AMAX-AMIN	010553
	DO 1 I=1,66	010554
	DO 1 J=1,3	010555
	T=2.**((J-2)*FNSQ*.1	010556
	IF (ADIF.LE.T*10.**((1-J3)) GO TO 2	010557
1	CONTINUE	010558
2	S=T*10.**((1-33)/FNSQ	010559
	IF (AMIN) 3,7,4	010560
3	AL=AL-S	010561
	IF (AL-AMIN) 7,7,3	010562
4	AL=AL+S	010563
	IF (AL-AMIN) 4,7,6	010564
6	AL=AL-S	010565
7	AU=AL+S*FNSQ	010566
	IF (AU.GE.AMAX) GO TO 11	010567
	IF (J.LE.3) GO TO 10	010568
	J=0	010569
	I=I+1	010570
10	J=J+1	010571
	T=2.**((J-2)*FNSQ*.1	010572
	AL=0.	010573
	GO TO 4	010574
11	AL=AL-10.*S	010575
	RETURN	010576
	END	010577

C	SUBROUTINE PLOTSS (YMAXIN,YMININ,YTOP,YBOT)	010578
C		010579
C	DEBUG # 94	010580
C	SUBROUTINE SELECTS PLOT UPPER AND LOWER LIMIT FOR A 10 SQUARE	010581
C	LINEAR PLOT GRID.	010582
C		010583
C	-----SUBROUTINE ARGUMENT DESCRIPTIONS-----	010584
C		010585
C	YMAXIN = INPUT MAXIMUM VALUE TO BE PLOTTED.	010586
C	YMININ = INPUT MINIMUM VALUE TO BE PLOTTED.	010587
C	YTOP = OUTPUT UPPER GRID LIMIT.	010588
C	YBOT = OUTPUT LOWER GRID LIMIT.	010589
C		010590
	DATA NUT/ 6 /	010591
	YMAX = YMAXIN	010592
	YMIN = YMININ	010593
		010594
	IF (YMAX .LT. YMIN) GO TO 999	010595
	IF (YMAX .GT. YMIN) GO TO 21	010596
11	IF (YMAX .LT. 0.00) GO TO 13	010597
	YMAX = 1.001*YMAX	010598
	YMIN = 0.999*YMIN	010599
	GO TO 15	010600
13	YMAX = 0.999*YMAX	010601
	YMIN = 1.001*YMIN	010602
15	IF (YMAX .NE. 0.0) GO TO 21	010603
	YMAX = +0.3	010604
	YMIN = -0.3	010605
C		010606
21	VALUE = (YMAX-YMIN)/10.	010607
	IF (VALUE .LT. ABS(YMIN/100000.)) GO TO 11	010608
	DO 23 I=1,66	010609
	DO 23 J=1,3	010610
	SCALE = 2.**((J-2) * 10. **((I-33)	010611
23	IF (SCALE .GE. VALUE) GO TO 31	010612
		010613
	GO TO 999	010614
C		010615
31	NSTEPS = YMIN/SCALE	010616
	YBOT = FLOAT(NSTEPS)*SCALE	010617
32	IF (YMIN) 34,38,36	010618
33	YBOT = YBOT-SCALE	010619
34	IF (YBOT .LE. YMIN) GO TO 38	010620
	GO TO 33	010621
35	YBOT = YBOT+SCALE	010622
36	IF (YBOT-YMIN) 35,38,37	010623
37	YBOT = YBOT-SCALE	010624
38	YTOP = YBOT+10.*SCALE	010625
	IF (YTOP .GE. YMAX) RETURN	010626
	IF (J .LT. 3) GO TO 39	010627
	J = 0	010628
	I = I+1	010629
39	J = J+1	010630
	SCALE = 2.**((J-2) * 10.**((I-33)	010631
	GO TO 32	010632
C		010633
999	WRITE (NLT,2001) NERROR	010634
2001	FORMAT (5X,49HEXAK ENCOUNTERED IN SUBROUTINE PLOTSS, NERROR = 13,	010635
	* /,5X, 16HPROGRAM STOPPED.)	010636
	STOP	010637
	END	010638

SUBROUTINE ALPHA (ALPHA,A,Z,NR,NC,KR)	010639
C	010640
IMPLICIT REAL*8(A-H,O-Z)	010641
DIMENSION A(KR,1), Z(KR,1)	010642
C	010643
C SCALAR ALPHA TIMES MATRIX A. (ALPHA * A = Z).	010644
C MATRICES A,Z MAY SHARE SAME CORE LOCATIONS.	010645
C CODED BY RL WUHLIN. FEBRUARY 1965.	010646
C	010647
C SUBROUTINE ARGUMENTS	010648
C ALPHA = INPUT SCALAR.	010649
C A = INPUT MATRIX. SIZE(NR,NC).	010650
C Z = OUTPUT RESULT MATRIX. SIZE(NR,NC).	010651
C NR = INPUT NUMBER OF ROWS IN MATRICES A,Z.	010652
C NC = INPUT NUMBER OF COLS IN MATRICES A,Z.	010653
C KR = INPUT ROW DIMENSION OF A,Z IN CALLING PROGRAM.	010654
C	010655
DO 10 I=1,NR	010656
DO 10 J=1,NC	010657
10 Z(I,J) = ALPHA * A(I,J)	010658
RETURN	010659
END	010660

SUBROUTINE BDOT (A,B,Z,NRB,NCB,KA,KB)	010661
C	010662
IMPLICIT REAL*8(A-H,O-Z)	010663
DIMENSION A(KA,1), B(KB,1), Z(KB,1)	010664
COMMON /LWRKVI/ W(100)	48C10665
C	010666
C SPECIAL TRIPLE MATRIX PRODUCT. B*A*B(TRANPOSE) = Z.	010667
C A MUST BE SYMMETRIC TO GET CORRECT ANSWER.	010668
C Z WILL BE SYMMETRIC. UPPER HALF CALCULATED, REFLECTED TO LOWER HALF.	010669
C THE MAXIMUM SIZE IS	010670
C NCB = 500	010671
C DEVELOPED BY CARL BUDLEY. JANUARY 1965.	010672
C LAST REVISION BY RL WUHLIN. JULY 1972.	010673
C	010674
C SUBROUTINE ARGUMENTS	010675
C A = INPUT INNER MATRIX. SIZE(NCB,NCB).	010676
C B = INPUT OUTER MATRIX. SIZE(NRB,NCB).	010677
C Z = OUTPUT RESULT MATRIX. SIZE(NRB,NRB).	010678
C NR = INPUT NUMBER OF ROWS OF MATRIX B. SIZE OF MATRIX Z.	010679
C NCB = INPUT NUMBER OF COLS OF MATRIX B. SIZE OF MATRIX A. MAX=500.	010680
C KA = INPUT ROW DIMENSION OF A IN CALLING PROGRAM.	010681
C KB = INPUT ROW DIMENSION OF B,Z IN CALLING PROGRAM.	010682
C	010683
C	010684
DO 40 J=1,NRB	010685

DO 20 L=1,NCB	010686
S = C.D0	010687
DO 10 K=1,NCB	010688
10 S = S + A(L,K)*B(J,K)	010689
20 W(L) = S	010690
DO 40 I=1,J	010691
S = C.D0	010692
DO 30 L=1,NCB	010693
30 S = S + B(I,L)*W(L)	010694
Z(I,J) = S	010695
40 Z(J,I) = S	010696
RETURN	010697
END	010698
SUBROUTINE MULTB (A,BZ,NRA,NRB,NCB,KA,KBZ)	010699
C	DEBUG # 97 010700
IMPLICIT REAL*8(A-H,O-Z)	010701
DIMENSION A(KA,1),BZ(KBZ,1)	010702
COMMON /LWRKVI/ W(100)	48010703
C	010704
C MATRIX MULTIPLICATION. A * B = Z.	010705
C USES TWO WORK SPACES. RESULT (Z) IS PLACED IN B.	010706
C BZ MUST BE DIMENSIONED LARGE ENOUGH IN MAIN PROGRAM TO CONTAIN THE	010707
C LARGER OF B OR Z.	010708
C THE MAXIMUM SIZE IS	010709
C NRB = 500.	010710
C	010711
C -----SUBROUTINE ARGUMENT DESCRIPTIONS-----	010712
C	010713
C A = INPUT MATRIX. SIZE (NRA,NRB).	010714
C BZ = INPUT MATRIX. SIZE (NRB,NCB).	010715
C = OUTPUT RESULT MATRIX. SIZE (NRA,NCB).	010716
C NRA = INPUT NUMBER OF ROWS OF MATRICES A,Z.	010717
C NRB = INPUT NUMBER OF ROWS OF MATRIX B, COLS OF MATRIX A. MAX =	010718
C NCB = INPUT NUMBER OF COLS OF MATRICES B,Z.	010719
C KA = INPUT ROW DIMENSION OF A IN CALLING PROGRAM.	010720
C KBZ = INPUT ROW DIMENSION OF BZ IN CALLING PROGRAM.	010721
C	010722
DO 40 J=1,NCB	010723
DO 20 K=1,NRB	010724
20 W(K) = BZ(K,J)	010725
DO 40 I=1,NRA	010726
S = C.D0	010727
DO 30 K=1,NRB	010728
30 S = S + A(I,K)*W(K)	010729
40 BZ(I,J) = S	010730
C	010731
RETURN	010732
END	010733

```

SUBROUTINE REVAAD (ALPHA,A,IVEC,JVEC,Z,NRA,NCA,NRZ,NCZ,KRA,KRZ) 010734
C                                                                    010735
C                                                                    010736
C                                                                    010737
C                                                                    010738
C                                                                    010739
C REARRANGE AND ADD ROWS AND COLUMNS OF ALPHA * MATRIX A INTO 010740
C MATRIX Z. 010741
C BE SURE MATRIX Z IS DEFINED BEFORE CALLING THIS SUBROUTINE. FOR 010742
C EXAMPLE, CALL ZERO TO CLEAR MATRIX Z. 010743
C 010744
C SUBROUTINE ARGUMENTS 010745
C ALPHA = INPUT SCALAR THAT MULTIPLIES MATRIX A. 010746
C A = INPUT MATRIX TO BE ARRANGED AND ADDED. SIZE(NRA,NCA). 010747
C IVEC = INPUT VECTOR. SIZE(NRA). 010748
C IVEC(I)=ROW POSITION OF A(ROW I) IN Z. 010749
C IF IVEC(I) IS PLUS ,Z=Z(ROW IVEC(I))+ALPHA*A(ROW I). 010750
C IF IVEC(I) IS MINUS,Z=Z(ROW IVEC(I))-ALPHA*A(ROW I). 010751
C IF IVEC(I) IS ZERO , A(ROW I) IS OMITTED IN Z. 010752
C JVEC = INPUT VECTOR. SIZE(NCA). 010753
C JVEC(J)=COL POSITION OF A(COL J) IN Z. 010754
C IF JVEC(J) IS PLUS ,Z=Z(COL JVEC(J))+ALPHA*A(COL J). 010755
C IF JVEC(J) IS MINUS,Z=Z(COL JVEC(J))-ALPHA*A(COL J). 010756
C IF JVEC(J) IS ZERO , A(COL J) IS OMITTED IN Z. 010757
C Z = INPUT/OUTPUT MATRIX TO WHICH ALPHA*A IS ADDED. SIZE(NRZ,NCZ). 010758
C NRA = INPUT NUMBER OF ROWS IN MATRIX A. 010759
C NCA = INPUT NUMBER OF COLS IN MATRIX A. 010760
C NRZ = INPUT NUMBER OF ROWS IN MATRIX Z. 010761
C NCZ = INPUT NUMBER OF COLS IN MATRIX Z. 010762
C KRA = INPUT ROW DIMENSION OF A IN CALLING PROGRAM. 010763
C KRZ = INPUT ROW DIMENSION OF Z IN CALLING PROGRAM. 010764
C 010765
C DO JC 1A=1,NRA 010766
C IZ = IABS(IVEC(1A)) 010767
C IF (IZ .EQ. 0) GO TO 30 010768
C DO 25 JA=1,NCA 010769
C JZ = IABS(JVEC(JA)) 010770
C IF (JZ .EQ. 0) GO TO 25 010771
C SIGN = +1.00 010772
C IF (IVEC(1A).LT.0 .AND. JVEC(JA).GT.0 .OR. 010773
C * IVEC(1A).GT.0 .AND. JVEC(JA).LT.0) SIGN=-1.00 010774
C Z(IZ,JZ) = Z(IZ,JZ) + SIGN*ALPHA*A(1A,JA) 010775
C 25 CONTINUE 010776
C 30 CONTINUE 010777
C RETURN 010778
C 010779
C 010780
C ENCL

```

```

SUBROUTINE LRESP 010781
C                                                                    010782
C                                                                    010783
C                                                                    010784
C                                                                    010785
C SUBROUTINE SOLVES FOR THE LINEARIZED TIME RESPONSE. 010786
C TECHNIQUE USES A RUNGE KUTTA STARTER AND AN ADAMS CORRECTOR 010787
C RECURSIVE FORMULA FOR THE TIME SOLUTION. 010788

```

C		C10788
C	FOR OUTLINE OF PROCEDURES USED HERE IN SEE VOL 1,PAGE 85	010789
C	' LINEAR TIME DOMAIN RESPONSE'	010790
C		010791
	COMMON /KDSIZE/	010792
1	KR, KRT, KRX, KV1, KV2, KVA	010793
	COMMON /LDSIZE/	010794
2	NX, NY, NOLTA, NXSS, NB, NUJ, NY2, ND2	010795
	COMMON /TAPEND/	010796
4	NUF1, NUF2, NUF3	010797
	COMMON /MISCND/	010798
5	NUPRNT, NCPLUT	010799
	COMMON / LV1 /	010800
C	V1 (100), V2 (100), V3 (100)	42610801
	COMMON /VECTOR/	010802
E	Y (250), YD (250)	43010803
	COMMON /LWURK1/	010804
F	W1(100,100), W2(100,100)	43210805
	COMMON /TIMESS/	010806
G	ST, DT, T, ET, TMST	010807
		010808
C	ASSUMES THAT W1 = A* ON ENTRY.	010809
C	UNIT NUF3 WILL BE WRITTEN FOR PLOTTING	010810
C		010811
	DIMENSION PRK(4), YDS(250,5), YS(250)	47010812
C		010813
C	STORE Y* IN YS, THEN ZERO Y.	010814
	DO 20 I=1,NX	010815
	YS(I) = Y(I)	010816
20	Y(I) = 0.00	010817
	PRK(1) = 0.500	010818
	PRK(2) = 1.00 - DSQRT(0.500)	010819
	PRK(3) = 1.00 + DSQRT(0.500)	010820
	PRK(4) = 0.500	010821
C		010822
	NT = 0	010823
	T = ST	010824
	TMST = 0.00	010825
	IPRNT = 0	010826
	IPLUT = 0	010827
C		010828
	REWIND NUF3	010829
C		010830
	WRITE (NUF3) ((W1(I,J),I=1,KR),J=1,KR)	010831
	REWIND NUF3	010832
C		010833
	CON = 3.00 * DT/8.00	010834
C		010835
	DO 30 I=1,NX	010836
	DO 30 J=1,NX	010837
	W1(I,J) = -CON*W1(I,J)	010838
	IF (J .EQ. 1) W1(I,J) = 1.+W1(I,J)	010839
30	CONTINUE	010840
C		010841
	CALL GAUSS1 (W1,W2,NX,KR)	010842
	READ (NUF3) ((W1(I,J),I=1,KR),J=1,KR)	010843
	REWIND NUF3	010844
C		010845
	CALL ZERO (V1,1,NX,1)	010846
	CALL LTRQL (V2)	010847

CALL YDCTL (W1,V2,Y,YD,NX,KK)	010848
C	010849
DO 10 I=1,NX	010850
10 YDS(I,1) = YD(I)	010851
C USE THE R-K STARTER.	010852
CALL LPRNT	010853
CALL LPLTWR	010854
C	010855
100 CONTINUE	010856
DO 120 J=1,4	010857
JIL = J	010858
DO 110 I=1,NX	010859
Z = YD(I) * DT	010860
GO TO (103,101,101,105),JIL	010861
101 R = PRK(JIL) * (Z-V1(I))	010862
GO TO 107	010863
103 R = PRK(JIL) * Z - V1(I)	010864
GO TO 107	010865
105 R = (Z-2.00 * V1(I)) / 0.00	010866
107 Y(I) = Y(I) + R	010867
110 V1(I) = V1(I) + 3.00 * R - PRK(JIL) * Z	010868
IF (JIL .EQ. 1 .OR. JIL .EQ. 3) T = T + 01/2.00	010869
CALL LTRCGL (V2)	010870
120 CALL YDCTL (W1,V2,Y,YD,NX,KK)	010871
C	010872
C	010873
NT = NT + 1	010874
ANT = NT	010875
TMST = ANT * DT	010876
T = ST + TMST	010877
C	010878
IPRNT = IPRNT + 1	010879
IF (IPRNT .NE. NOPRNT) GO TO 130	010880
CALL LPRNT	010881
IPRNT = 0	010882
130 CONTINUE	010883
IPLOT = IPLOT + 1	010884
IF (IPLOT .NE. NOPLOT) GO TO 140	010885
CALL LPLTWR	010886
IPLOT = 0	010887
140 CONTINUE	010888
DO 150 I=1,NX	010889
150 YDS(I,NT+1) = YD(I)	010890
IF (T .LE. ET .AND. NT .LT. 2) GO TO 100	010891
C	010892
C THE ADAMS CORRECTOR FORMULA	010893
C	010894
C0 = DT / 24.00	010895
C1 = C0 * 9.00	010896
C2 = C0 * 19.00	010897
C3 = -C0 * 5.00	010898
C4 = C0	010899
C	010900
C	010901
C ESTABLISH Y AT STEP NT	010902
C	010903
200 CALL LTRCGL (V1)	010904
C	010905
C V1 IS EXTERNAL FORCING FUNCTION FOR THE LINEAR SYSTEM.	010906
C	010907

CO 210 I=1,NX	010908
210 Y(I) = Y(I) + C1*V1(I) + C2*YDS(I,3) + C3*YDS(I,2) + C4*YDS(I,1)	010909
CALL MULTB (W2,Y,NX,NX,1,KR,KR)	010910
C	010911
C RESET YDS FOR NEXT STEP.	010912
C	010913
CO 220 I=1,NX	010914
YDS(I,1) = YDS(I,2)	010915
220 YDS(I,2) = YDS(I,3)	010916
C	010917
C COMPUTE YD AT STEP NT.	010918
CALL YDCTL (W1,V1,Y,YD,NX,KR)	010919
C	010920
CO 225 I=1,NX	010921
225 YDS(I,3) = YD(I)	010922
C	010923
NT = NT + 1	010924
ANT = NT	010925
TMST = ANT * DT	010926
T = ST + TMST	010927
C	010928
IPRNT = IPRNT + 1	010929
IF (IPRNT .NE. NIPRNT) GO TO 230	010930
CALL LPRNT	010931
IPRNT = 0	010932
230 CONTINUE	010933
IPLUT = IPLUT + 1	010934
IF (IPLUT .NE. NOPLUT) GO TO 240	010935
CALL LPLTWR	010936
IPLUT = 0	010937
240 CONTINUE	010938
C	010939
IF (1 .LT. ET) GO TO 200	010940
C	010941
C	010942
C	010943
RETURN	010944
END	010945
SUBROUTINE LPLTWR	010946
C	010947
IMPLICIT REAL*8(A-H,O-Z)	010948
C	010949
C SUBROUTINE WRITES TAPE NUT3 FOR PLOTTING.	010950
C	010951
COMMON /LDSIZE/	010952
2 NX, NY, NDLTA, NXSS, NB, NJU, NY2, NU2	010953
COMMON /TAPENO/	010954
4 NUT1, NUT2, NUT3	010955
COMMON /VECTOR/	010956
E Y (250), YD (250)	43010957
COMMON /TIMESS/	010958
G ST, DT, T, ET, TMST	010959
COMMON /PLTDTA/	010960
I NRPLUT, NCPLUT	010961
	DEBUG # 100

DATA IIST / 0 /	010962
IF (IIST .NE. 0) GO TO 5	010963
IIST = 1	010964
REWIND NLT3	010965
NRFLCT = 0	010966
NCFLCT = 2*NX + 1	010967
5 NRFLCT = NRFLCT + 1	010968
WRITE (NLT3) T.(YD(1),I=1,NX).(Y(1),I=1,NX)	010969
C	010970
RETURN	010971
END	010972
SUBROUTINE LPRNT	010973
C	010974
IMPLICIT REAL*8(A-H,O-Z)	010975
C	010976
C SUBROUTINE PRINTS OUT RESULTS OF LINEARIZED TIME RESPONSE.	010977
C	010978
COMMON /VECTOR/	010979
E Y (250), YD (250)	43010980
COMMON /LDSIZE/ NX,NY,NLTA,NXSS,NB,NJO,NY2,NU2	010981
COMMON /TIMESS/	010982
G ST, DT, T, LT, TMST	010983
DATA NLT/ 0 /	010984
DATA IIST/ 0 /	010985
C	010986
IF (IIST .NE. 0) GO TO 5	010987
IIST = 1	010988
C	010989
C PRINT OUT DATA AT START.	010990
C	010991
CALL PAGEHD	010992
WRITE (NCT,11)	010993
11 FORMAT (////30X,24HLINEARIZED TIME RESPONSE /	010994
* 32X,24H GENERAL INFORMATION //	010995
WRITE (NCT,12) ST,DT,ET	010996
12 FORMAT (//30X,26HINTEGRATION PARAMETERS ARE //	010997
* 30X,15HSTART TIME = 012.5/,	010998
* 30X,15HDELTA TIME = 012.5/,	010999
* 30X,15HEND TIME = 012.5)	011000
C	011001
LY = 1	011002
LX = LY+NY2	011003
LD=LX+NXSS	011004
LB=LD+NU2	011005
C	011006
5 CONTINUE	011007
CALL PAGEHD	011008
WRITE (NCT,101) T	011009
101 FORMAT (/30X,18HSIMULATION TIME = 012.5)	011010
C	011011
WRITE (NCT,102) NY2,NXSS,NU2,NU	011012
102 FORMAT (/30X,30HNUMBER OF PLANT VARIABLES = 15,	011013
* /30X,30HNUMBER OF SENSOR SIGNALS = 15,	011014
* /30X,30HNUMBER OF CONTROL VARIABLES = 15,	011015

	* /30X,30HNUMBER OF CONTROL TORQUES = 15)	011016
C	WRITE (NCT,103)	011017
	103 FORMAT (//20X,29HSTATE VECTOR TIME DERIVATIVES /)	011018
	CALL WRITES (YD,1,NX,1)	011019
C		011020
	WRITE (NCT,104)	011021
	104 FORMAT (//20X,12HSTATE VECTOR /)	011022
	CALL WRITES (Y,1,NX,1)	011023
C		011024
	WRITE (NCT,105)	011025
	105 FORMAT (//20X,20HPLANT VARIABLE RATES)	011026
	CALL WRITES (YD,1,NY2,1)	011027
C		011028
	WRITE (NCT,106)	011029
	106 FORMAT (//20X,19HSENSOR SIGNAL RATES)	011030
	CALL WRITES (YD(LX),1,NXSS,1)	011031
C		011032
	WRITE (NCT,107)	011033
	107 FORMAT (//20X,22HCONTROL VARIABLE RATES)	011034
	CALL WRITES (YD(LD),1,ND2,1)	011035
C		011036
	WRITE (NCT,108)	011037
	108 FORMAT (//20X,19HTORQUE SYSTEM RATES)	011038
	CALL WRITES (YD(LB),1,NB,1)	011039
C		011040
	WRITE (NCT,109)	011041
	109 FORMAT (///20X,20HPLANT VARIABLE STATE)	011042
	CALL WRITES (Y,1,NY2,1)	011043
C		011044
	WRITE (NCT,110)	011045
	110 FORMAT (//20X,19HSENSOR SIGNAL STATE)	011046
	CALL WRITES (Y(LX),1,NXSS,1)	011047
C		011048
	WRITE (NCT,111)	011049
	111 FORMAT (//20X,22HCONTROL VARIABLE STATE)	011050
	CALL WRITES (Y(LD),1,ND2,1)	011051
C		011052
	WRITE (NCT,112)	011053
	112 FORMAT (//20X,19HTORQUE SYSTEM STATE)	011054
	CALL WRITES (Y(LB),1,NB,1)	011055
C		011056
	RETURN	011057
	END	011058
		011059
	SUBROUTINE YDOTL (A,B,Y,YD,NY,KA)	011060
C		011061
	IMPLICIT REAL*8(A-H,O-Z)	011062
C		011063
C	SUBROUTINE FORMS THE LINEARIZED YDOT VECTOR.	011064
C		011065
C	-----SUBROUTINE ARGUMENT DESCRIPTIONS-----	011066
C		011067
C	A = INPUT LINEARIZED COEFFICIENTS. SIZE NA BY NY.	011068
C	USED IN EXPRESSION YD = A * Y + B	011069

C	B	= INPUT EXTERNAL FORCING TORQUES. USER SUPPLIED VIA	011070
C		SUBROUTINE LTORQL.	011071
C	Y	= INPUT VECTOR OF STATE VARIABLES.	011072
C	YD	= OUTPUT VECTOR OF STATE VECTOR TIME DERIVATIVES.	011073
C	NY	= SIZE OF STATE VECTOR TO BE INTEGRATED.	011074
C			011075
		DIMENSION A(KA,1),B(1),Y(1),YD(1)	011076
C			011077
		DO 10 I=1,NY	011078
		YD(I) = B(I)	011079
		DO 10 J=1,NY	011080
		YD(I) = YD(I) + A(I,J) * Y(J)	011081
	10	CONTINUE	011082
		RETURN	011083
		END	011084

		SUBROUTINE LTORQL (VTORQ)	011085
C		DEBUG # 103	011086
		IMPLICIT REAL*8(A-H,O-Z)	011087
C		***** USER SUPPLIED SUBROUTINE *****	011088
C			011089
C		VTORQ ARRAY TO CONTAIN THE ELEMENTS B IN THE EQUATION	011090
C		Y(OUT) = A*Y + B	011091
C		THE LINEARIZED EQUATIONS OF MOTION	011092
C			011093
		DIMENSION VTORQ(1)	011094
		COMMON /KDSIZE/	011095
	I	KR, KRT, KRX, KVI, KV2, KVX	011096
		COMMON /VECTOR/	011097
	E	Y (250), YD (250)	43011098
		COMMON /TIMESS/	011099
	G	ST, DT, T, ET, TMST	011100
C			011101
C			011102
		CALL ZERO (VTORQ,I,KVX,1)	011103
		RETURN	011104
		END	011105

		FUNCTION ADT (IC,T)	011106
C		DEBUG # 104	011107
C		USER DEFINED FUNCTION	011108
C		USED TO DEFINE THE ALPHA DOT TERM IN EQUATION 11-5 VOL 1	011109
C		THE VELOCITY OF THE COORDINATE IC WHICH IS RHEUMICALLY	011110
C		CONSTRAINED IS DEFINE HERE. SEE SUBROUTINE FINDU	011111
C		SEE ADT GET THIS TERM VIA INTEGRATION OF ADD1 IN CONTRL	011112
C		NUMERICAL PROBLEMS MAY ARISE IF THIS IS NOT DONE	011113
		IMPLICIT REAL*8 (A-H,O-Z)	011114
		COMMON /VECTOR/	011115
	*	Y(250),YDT(250)	2011116
C			011117

C	RETURN	011118
	END	011119
		011120
	FUNCTION ADDT (IC,T)	011121
C		011122
C	USER DEFINED FUNCTION	011123
C	USED TO DEFINE ACCELERATION OF RHEOMICALLY CONSTRAINED	011124
C	COORDINATE IC, SEE SUBROUTINE YDOT	011125
C	THE ALPHA DOUBLE DOT TERM IN EQUATION 11-6	011126
C	CAUTION INTEGRATE ADDT IN CNTRL TO GET ADT.	011127
C	STEP CHANGES IN VELOCITY NUMERICALLY LEAD TO EXTRANEIOUS	011128
C	IMPULSES	011129
C	IMPLICIT REAL*8 (A-H,O-Z)	011130
	COMMON /VECTOR/	011131
C	* Y(250),YDT(250)	2011132
	RETURN	011133
	END	011134
		011135
	SUBROUTINE KHINGE (G)	011136
C		011137
	IMPLICIT REAL*8 (A-H,O-Z)	011138
	DIMENSION G(1)	011139
C		011140
	COMMON /SHOSRD/	011141
	* BH(6,18,11),BS(6,18,15),ROL(3,3, 6),JOL(3, 6)	211142
	COMMON /CONPAR/	011143
	* CNTCTA(150)	9511144
	COMMON /MAXMUM/	011145
	* NEMAX,NHMAX,NSPMAX,NMWMAX,NMWEOO,NMOBJO,KMU,KY,KU	011146
	COMMON /MUMENG/	011147
	* P(113),PMOM(36),HTOT(3),TOTL(3),ENGKE(6),ENGPE(6),	1111148
	* TOTKE, TOTPE, TOTENG, AHTCT,ATOTL	011149
	COMMON /SPLCIF/	011150
	* BETAH(6, 6),BETAHO(6, 6),AMO(2, 5),RH(3,3,30),RS(3,3,30),	1611151
	* CH(3,35),DS(3,30),IMO(3, 5),NMOW(6, 6),IFISMW(15),	1711152
	* NO,NH,NSPT,NUFMO,NOELTA,ITOPOL(2, 6),IRGFLX(6),IHDATA(7, 6),	1811153
	* LOCU(14),LENU(14),NU,NBETA,NLAM,NEQ	1911154
C		011155
	DIMENSION SK(3,6),OK(3,6)	011156
	DIMENSION HNGT(6, 6)	9211157
C	ZERO OUT HNGT ARRAY AND POTENTIAL ENERGY TERM	011158
	DO 11 L=1,NH	011159
	DO 11 I=1,6	011160
	11 HNGT(I,L) = 0.00	011161
	TOTPE = 0.00	011162
C		011163
C		011164
	C***** USER	011165

C	10 SET UP HINGE TORQUES DUE TO SPRINGS AND DASHPOTS ACROSS	011166
C	GIMBAL AXES	011167
C	1) EQUIVALENCE ARRAYS WHICH HOLD CONSTANTS TO PROPER AREA	011168
C	OF DATA BLOCK CNTDTA	011169
C	2) GET HINGE TORQUE ABOUT EACH AXIS	011170
C	3) COMPUTE POTENTIAL ENERGY	011171
C	EXAMPLE	011172
C	EQUIVALENCE (CNTDTA(12),SK(1)), (CNTDTA(30),JK(1))	011173
C	DO 10 L=1,NH	011174
C	DO 10 I=1,3	011175
C	FNGT(I,L) = -(SK(I,L)*BETAH(I,L) + DK(I,L)*BETAHD(I,L))	011176
C	10 TGTPE = TOTPE + 0.500*SK(I,L)*BETAH(I,L)**2	011177
C		011178
C	ADD HINGE TORQUE/FORCE EFFECTS INTO BODY 1 G VECTOR	011179
C	LEQ = IRGFLX(1) + 6	011180
C	DO 15 I=1,6	011181
C	F = FNGT(I,1)	011182
C	DO 15 J=1,LLQ	011183
C	15 G(J) = G(J) + F*BH(I,J,1)	011184
C	15 CONTINUE	011185
C		011186
C	ADD HINGE TORQUE/FORCE EFFECTS TO G VECTOR FOR EACH HINGE	011187
C	DO 20 L=2,NH	011188
C	NOBQ = ITOPOL(1,L)	011189
C	NOBP = ITOPOL(2,L)	011190
C	LQ = 2*L - 2	011191
C	LP = LQ + 1	011192
C	LOQ = LCCJ(NOBQ) - 1	011193
C	LCP = LCCJ(NOBP) - 1	011194
C	LEQ = IRGFLX(NOBQ) + 6	011195
C	LEP = IRGFLX(NOBP) + 6	011196
C	DO 20 I=1,6	011197
C	F = FNGT(I,L)	011198
C	DO 25 J=1,LEQ	011199
C	LOQJ = LOQ + J	011200
C	25 G(LOQJ) = G(LOQJ) + F*BH(I,J,LO)	011201
C	DO 25 J=1,LEP	011202
C	LCPJ = LCP + J	011203
C	26 G(LCPJ) = G(LCPJ) + F*BH(I,J,LP)	011204
C	20 CONTINUE	011205
C		011206
C	RETURN	011207
C	END	011208
C	SUBROUTINE CONTROL	011209
C	IMPLICIT REAL*8 (A-H,O-Z)	011210
C	DEBUG # 107	011211
C	SUBROUTINE CONTROL MUST BE USER DEFINED	011212
C	IT IS USED TO ESTABLISH THE TIME DERIVATIVES OF THE CONTROL	011213
C	SYSTEM STATE VARIABLES	011214
C		011215
C	COMMON /JHESRO/	011216
C	* BH(6,18,11),BS(6,18,15),ROL(3,3, 6),JUL(3, 6)	211217
C	COMMON /CONPAR/	011218
C	* CNTDTA(100)	9511219

```

COMMON /LDSIZE/ NX,NY,NOLTA,NXSS,NBTQ,NJQ,NY2,NDZ      011220
COMMON /SPECIF/                                         011221
*   BETAH(6, 6),BETAHD(6, 6),AMO(2, 5),RH(3,3,30),RS(3,3,30), 1611222
*   CH(3,35),DS(3,30),IMO(3, 5),NMOW(6, 5),IFISM(15),      1711223
*   NB,NH,NSPT,NUFMO,NDELTA,ITOPOL(2, 6),IRGFLX( 6),IHDATA(7, 6), 1811224
*   LOCU(14),LENU(14),NU,NBETA,NLAM,NEQ                  1911225
COMMON /TIMES/                                          011226
*   STARTI,DELTAT,T,ENDT,IMST                             011227
COMMON /VECTOR/                                         011228
*   Y(250),YDT(250)                                       2011229
CCCCCCC THIS COMMON IS TRANSFER BETWEEN CNTRL AND SHAFT ONLY ---- 011230
C   COMMON /WHEEL /                                         011231
C   *   CLM(4)                                              011232
C   IF DATA IS TO BE COMPUTED WHICH MUST BE TRANSFERED TO OTHER 011233
C   USER DEFINED SUBROUTINES ADD ADDITIONAL SPECIAL PURPOSE     011234
C   COMMON BLOCKS                                              011235
C   FUNCTIONS REQUIRED BY SAMPLE PROBLEM                       011236
C   ALIM(U,V) = UMAXI(-V,UMINI(U,V))                          011237
C   ADD DIMENSION STATEMENTS FOR CONTROL SYSTEM DESCRIPTION     011240
C   DIMENSION TQ(6),TGD(6),RHD(3),THADW(3)                    011241
C   DATA ICT4/0/, RHD / 0.00, 0.00, 0.00 /                   011242
C   SET UP DIMENSION STATEMENTS AND DEFAULT VALUES FOR TRANSFER 011244
C   FUNCTION INPUT DIRECTLY INTO CNTRL                         011245
C   NPLY = NUMBER OF POLYNOMIAL RATIO PAIRS (NUMERATOR AND      011246
C   DENOMIATOR) TO BE READ ON FIRST PASS THRU CNTRL            011247
C   KRY = ROW DIMENSION OF CPLY (HIGHEST DENOMINATOR ORDER + 1 011248
C   KCY = COLUMN DIMENSION OF CPLY (2 * NUMBER OF POLYNOMIAL    011249
C   TRANSFER FUNCTIONS TO BE READ)                              011250
C   CPLY(1,2*K-1) = DENOMINATOR COEFFICIENTS IN ASCENDING      011251
C   ORDER FOR INPUT POLY K, I=1,2,...,KPLY(K)                  011252
C   KPLY(K)-1 = DEGREE OF POLY TRANSFER FUNCTION DENOMINATOR   011253
C   ALSO NUMBER OF FIRST ORDER DIFFERENTIAL                   011254
C   EQUATIONS SET UP IN TPLY FOR K-TH TRANSFER                  011255
C   FUNCTION                                                      011256
C   CPLY(1,2*K) = NUMERATOR COEFFICIENTS ASCENDING ORDER       011257
C   SEE SUBROUTINE DEF3,'SUBROUTINE DYN520 INPJ1...DATA'        011258
C   DIMENSION CPLY(10,4), KPLY(2), UI(2)                       011259
C   DATA NPLY, KRY, KCY/ 0, 10, 4/                             011260
C   DATA I1ST/ 0 /                                             011261
C   CCCCCCCCCC                                                011262
C   CCCCCCCCCC                                                011263
C   THE FOLLOWING STATEMENTS MUST ALWAYS BE IN CNTRL..         011264
C   FIRST PASS THROUGH READ IN ALL CONTROL POLY TRANSFER FUNCTIONS 011265
C   IF (I1ST.NE. 0) GO TO 110                                   011266
C   I1ST = 1                                                    011267
C   IF (NPLY.EQ. 0) GO TO 106                                   011268
C   CALL ZERO (CPLY,KRY,KCY,KRY)                                011269
C   DO 105 K=1,NPLY                                             011270
C   K2=2*K-1                                                    011271
C   105 CALL READ (CPLY(1,K2),KPLY(K),K2,KRY,KCY)              011272
C   CALL WRITE (CPLY,KRY,KCY,4HCPLY,KRY)                       011273
C   106 CONTINUE                                                011274
C   NDLTA = NDELTA                                              011275
C   LDEL = LCCU(2*NB+2) - 1                                    011276
C   110 CONTINUE                                                011277

```

C	DEFAULT VALUES	011280
C	INTERNAL DO LOOP LOGIC REQUIRES THAT	011281
C	NAXX = NXSS + NBTQ	011282
C	WHERE	011283
C	NAXX > 0	011284
C	AND IT MUST BE REDEFINED IN EQADD	011285
C	NXSS = 1	011286
C	NBTQ = 0	011287
C	NXSS = NUMBER OF PLANT SENSOR SIGNALS	011288
C	NBTQ = NUMBER OF CONTROL SYSTEM OUTPUTS	011289
C	THESE MUST BE USER DEFINED IF NON-ZERO. FOR EXAMPLE	011290
C	NXSS = 2	011291
C	NBTQ = 3	011292
C	IF (NDELTA .EQ. 0) RETURN	011293
C	CCCCCCCCCCCC C	011294
C	CCCC---NOTE---THIS SUBROUTINE MUST ESTABLISH NDELTA,NXSS AND NBTQ	011295
C	CCCCCCCCCCCC	011296
C	CCCC ESTABLISH THE D/DOT(DELTAS)	011297
C	CCCCCCCCCCCC	011299
C	CCCC---NOTE---THIS SECTION IS TYPICAL OF USE OF TPLY.	011300
C	CCCCCCCCCCCC	011301
C	IF (NPLY .EQ. 0) GO TO 116	011302
C	L = LDEL+1	011303
C	L = STARTING LOCATION FOR FIRST DELTA OUT EQUATION	011304
C	GO 115 K=1,NPLY	011305
C	K2 = 2*K-1	011306
C	CALL TPLY TO SETUP DIFFERENTIAL EQUATIONS ASSOCIATED WITH POLY	011307
C	TRANSFER FUNCTIONS	011308
C	CALL TPLY (CPLY(1,K2),CPLY(1,K2+1),UI(K),X,KPLY(K),L)	011309
C	L = L+KPLY(K) - 1	011310
C	115 CONTINUE	011311
C	KPLY(1) + KPLY(2) + ... + KPLY(NPLY) - NPLY = THE NUMBER OF	011312
C	FIRST ORDER EQUATIONS DEFINED AND PUT IN YOT ARRAY	011313
C	116 CONTINUE	011314
C	CCCCCCCCCCCC	011315
C	EXAMPLE CONTROL SYSTEM FOR ATS-F	011316
C	ICT4 = ICT4 + 1	011317
C	IA = (ICT4-1)/4	011318
C	IAA = (ICT4-2)/4	011319
C	IFLAG = IA - IAA	011320
C	DO 6 I=1,3	011321
C	6 THADV(I) = Y(6+I)	011322
C	DO 5 I=1,6	011323
C	5 TQ(I) = Y(LDEL+I)	011324
C	WHEEL 1 (ROLL INERTIA WHEEL CONTROL TORQUE)	011325
C	DEFINE DIFFERENTIAL EQUATIONS FOR ROLL CONTROL LOOP	011326
C	U1 = 37.295800*ROL(3,2,1)/ROL(3,3,1)	011327
C	U5 = ALIM(TQ(5),29.00)	011328
C	U2 = 2.1700*U1 - U5	011329
C	U3 = ALIM(1.100*U2,1.1700)	011330
C	TQC(5) = (1.00/88.00)*(-TQ(5) + (9/1.100)*U3)	011331
C	U6 = ALIM(5*U3,1.6800)	011332
C	U8 = ALIM(TQ(6),1.900)	011333
C	IF (IFLAG .EQ. 0) GO TO 32	011334
C		011335
C		011336
C		011337
C		011338
C		011339

C	CU = CABS(U8)	011340
C	IF (CU.GT.1.00) GO TO 30	011341
C	IF (CU.LT.0.500) GO TO 31	011342
C	C9 = RHD(1)	011343
C	GO TO 10	011344
C	30 C9 = U8/CU	011345
C	GO TO 10	011346
C	31 C9 = 0.00	011347
C	GO TO 10	011348
C	32 C9 = RHD(1)	011349
C	GO TO 33	011350
C	10 RHD(1) = U9	011351
C	33 CONTINUE	011352
C	TQC(6) = (-TQ(6) + 2.500*(U6-U9))/.500	011353
C		011354
C	1500 RPM = 157.0795 RAD/SEC	011355
C	6 INCH*OZ = .03125 FT*LB5	011356
C		011357
C	IF (CABS(THADW(1)).GT. 157.079500) U9 = 0.00	011358
C	CLM(1) = .0312500*U9 - 5.0-U9*THADW(1)	011359
C		011360
C	WHEEL 2 (PITCH INERTIA WHEEL CONTROL TORQUE)	011361
C	DEFINE DIFFERENTIAL EQUATIONS IN PITCH CONTROL LOOP	011362
C		011363
C	U1 = -57.295800*ROL(3,1,1)/ROL(3,3,1)	011364
C	U5 = ALIM(TQ(1),16.400)	011365
C	U2 = 2.1700*U1 - U5	011366
C	U3 = ALIM(.8200*U2,1.1700)	011367
C	TQD(1) = (-TQ(1) + U3*(7/.8200))/50.00	011368
C	U6 = ALIM(5*U3,1.6800)	011369
C	U8 = ALIM(TQ(2),1.900)	011370
C	IF (IFLAG.EQ.0) GO TO 14	011371
C	CU = CABS(U8)	011372
C	IF (CU.GT. 1.00) GO TO 15	011373
C	IF (CU.LT.0.500) GO TO 16	011374
C	C9 = RHD(2)	011375
C	GO TO 12	011376
C	15 C9 = U8/CU	011377
C	GO TO 12	011378
C	16 C9 = 0.00	011379
C	GO TO 12	011380
C	14 C9 = RHD(2)	011381
C	GO TO 13	011382
C	12 RHD(2) = U9	011383
C	13 CONTINUE	011384
C	TQC(2) = (-TQ(2) + 2.500*(U6 - U9))/.500	011385
C	IF (CABS(THADW(2)).GE. 157.079500) U9 = 0	011386
C	CLM(2) = .0312500*U9 - 5.0-U9*THADW(2)	011387
C		011388
C	WHEEL 3 (YAW INERTIA WHEEL CONTROL TORQUE)	011389
C	DEFINE DIFFERENTIAL EQUATIONS FOR YAW CONTROL LOOP	011390
C		011391
C	U1 = 57.295800*ROL(2,1,1)/ROL(2,2,1)	011392
C	U2 = ALIM(U1,2.00)	011393
C	U6 = ALIM(TQ(3),29.00)	011394
C	U3 = 2.1700*U2 - U6	011395
C	U4 = ALIM(1.4700*U3,1.1700)	011396
C	TQC(3) = (1.00/88.00)*(-TQ(3) + (9/1.4700)*U4)	011397
C	U7 = ALIM(5*U4,1.6800)	011398
C	U9 = ALIM(TQ(4),1.900)	011399

```

C      IF (IFLAG.EQ.0) GO TO 20                                011400
C      LU = CABS(U9)                                           011401
C      IF (LU.GT.1.D0) GO TO 21                                011402
C      IF (LU.LT. 0.500) GO TO 22                              011403
C      U10 = RHD(3)                                           011404
C      GO TO 18                                                 011405
C 21 U10 = U9/100                                             011406
C      GO TO 18                                                 011407
C 22 U10 = U.D0                                              011408
C      GO TO 18                                                 011409
C 20 U10 = RHD(3)                                           011410
C      GO TO 24                                                 011411
C 18 RHD(3) = U10                                           011412
C 24 CONTINUE                                              011413
C      TQ(4) = (-TQ(4) + 2.500*(U7 - U10))/.500             011414
C      IF (CABS(THAUW(3)) .GT. 157.079500) U10 = 0.00       011415
C      CLM(3) = .0312500*U10 - 5.D-05*THAUW(3)             011416
C                                                              011417
C      PCT ALL CONTROL SYSTEM EQUATION IN YDT ARRAY         011418
C      DO 34 I=1,6                                           011419
C 34 YDT(LDEL+I) = TQ(I)                                       011420
C      YDT(LDEL+7) = Y(16)                                    011421
C      SK4 = CNTOTA(NDELTA+1)                                011422
C      DK4 = CNTOTA(NDELTA+2)                                011423
C      CLM(4) = -(SK4*Y(LDEL+7) + DK4*YDT(LDEL+7))          011424
C                                                              011425
C      RETURN                                                011426
C      END                                                    011427

```

```

SUBROUTINE EXTOR (TEX,ISPN,NTEX)                                011428
C                                                              011429
C      IMPLICIT REAL*8 (A-H,O-Z)                             011430
C      DIMENSION TEX(6,1), ISPN(1)                           011431
C                                                              011432
C      COMMON /MAXIMUM/                                         011433
C      *      NIMAX,NHMAX,NSPMAX,NMWMAX,NMWBD0,NMDD00,K40,KY,KU 011434
C      COMMON /SPECIF/                                          011435
C      *      BETAH(6, 6),BETAMD(6, 6),AMD(2, 5),RH(3,3,30),RS(3,3,30), 1611436
C      *      .CH(3,30),DS(3,30),IMC(3, 5),NM9A(6, 6),IFISM(15), 1711437
C      *      NS,NH,NSPT,NH,MU,NDELTA,I1UPGL(2, 6),IRUFLEX( 6),IHDATA(7, 6), 1811438
C      *      LCCL(14),LENG(14),NC,NDELTA,NLAM,NEO           1911439
C      COMMON /VECTOR/                                         011440
C      *      Y(250),YDT(250)                                2011441
C                                                              011442
C      DATA IIST / 0 /                                       011443
C                                                              011444
C      CCC ESTABLISH THE EXTERNAL FORCE/TORQUE (6-LONG VECTOR) AND NUMBER 011445
C      CCC THE CORRESPONDING SENSOR POINTS. ALSO ESTABLISH THE NUMBER OF 011446
C      CCC SIX-LONG VECTORS (NTEX).                            011447
C      SEE VOL 11, PAGE 12                                     011448
C      ISPN = INTEGER ARRAY TO DEFINE WHICH SENSOR POINTS ARE TO BE 011449
C      USED FOR FORCE/TORQUE INPUTS                             011450
C                                                              011451
C      IF (IIST .EQ. 1) GO TO 5                                011452
C      IIST = 1                                                011453

```

	DO 10 I=1,0	011454
	DO 10 J=1,NSPMAX	011455
	10 TEX(1,J) = 0.0 0	011456
C		011457
	5 NTEX = 0	011458
C		011459
	RETURN	011460
	END	011461

	SUBROUTINE SHAFTT (TSHFT)	011462
C		011463
	IMPLICIT REAL*8 (A-H,O-Z)	011464
	DIMENSION TSHFT(1)	011465
C		011466
	COMMON /MAXMUM/	011467
*	NSMAX,NHMAX,NSPMAX,NMWMAX,NMWBD0,NMDS00,KMU,KY,KU	011468
	COMMON /SPECIF/	011469
*	BETAH(6, 6),BETAHD(6, 6),AMO(2, 5),RH(3,3,30),RS(3,3,30),	1611470
*	DR(3,35),DS(3,30),IMU(3, 5),NMGW(6, 6),IFTSMW(15),	1711471
*	NB,NH,NSPT,NDFMO,NDELTA,ITOPOL(2, 6),IRGFLX(6),IHDATA(7, 6),	1811472
*	LUCU(14),LENU(14),NU,NBETA,NLAM,NEQ	1911473
	COMMON /VECTOR/	011474
*	Y(250),YDT(250)	2011475
C		011476
	DATA I1ST / 0 /	011477
C		011478
	IF (I1ST.EQ. 1) GO TO 20	011479
	I1ST = 1	011480
	DO 5 I=1,NMWMAX	011481
	5 TSHFT(I) = 0.0 0	011482
	20 CONTINUE	011483
C		011484
C	INTERFACE WITH SUBROUTINE CONTRL, WHEEL TORQUES COMPUTED	011485
C	IN CONTRL STORED IN CLM ARRAY, DISCOS VELDS THEM IN	011486
C	THE TSHFT ARRAY	011487
C	EXAMPLE:	011488
C	1) SETUP COMMON BLOCK TO TRANSFER DATA	011489
C	2) EQUATE TORQUE ARRAY CLM TO TSHFT TORQUE ARRAY	011490
C		011491
	CCCCCCC THIS COMMON IS TRANSFER BETWEEN CONTRL AND SHAFTT ONLY ----	011492
C	COMMON /WHEEL /	011493
C	* CLM(4)	011494
C	10 DO 15 I=1,4	011495
C	15 TSHFT(I) = CLM(I)	011496
C		011497
	RETURN	011498
	END	011499

	SUBROUTINE EQADD	011500
C		011501
	DEBUG # 110	

```

C      IMPLICIT REAL*8 (A-H,O-Z)                                011502
C      SEE VOL 11, PAGE 15                                       011503
C      AUXILIARY EQUATIONS WHICH DEFINE THE PHYSICALLY REALIZABLE 011504
C      SENSOR SIGNALS AND CONTROL TORQUES ARE TO BE LOADED IN THE 011505
C      YDT ARRAY AFTER THE LAST EQUATION. THAT IS YDT(NEQ)       011506
C      THE ORDER ASSUMED IS SENSOR SIGNAL, TORQUE SIGNAL        011507
C      COMMON /OBSRQ/                                           011508
C      *   BT(2,18,11),BS(2,18,15),ROL(3,3, 6),JUL(3, 6)      011509
C      COMMON /ONAXX /                                           011510
C      *   NAXX                                                    011511
C      COMMON /LOSIZE/ NX,NY,NJLTA,NXSS,NBTQ,NJO,YYZ,NBZ        011512
C      COMMON /MAXIMUM/                                           011513
C      *   NIMAX,NHMAX,NSPMAX,NMWMAX,NMWPOD,NMDBOD,NXJO,NY,KU     011514
C      COMMON /SPECIF/                                           011515
C      *   ELTA(6, 6),BELTA(6, 6),AMD(2, 5),RH(3,3,30),RS(3,3,30), 1611517
C      *   CH(3,35),DS(3,30),IMU(3, 5),NMGW(6, 5),IFTSW(15),    1711518
C      *   NG,NH,NSP,T,NCFMU,NDELTA,IICPOL(2, 6),IRGFLA( 6),IHDATA(7, 6), 1811519
C      *   LCCU(14),LENU(14),NG,NBETA,NLAM,NEQ                  1911520
C      COMMON /VECTOR/                                           011521
C      *   Y(250),YDT(250)                                       2011522
C
C      THIS SUBROUTINE MUST DEFINE NAXX                          011523
C      NAXX = NUMBER OF AUXILIARY EQUATIONS, DEFAULT = 1        011524
C      FOR THE LINEARIZATION PORTION OF DISCOS AT LEAST ONE     011525
C      AUXILIARY EQUATION REQUIRED. IT MUST BE A FUNCTION        011526
C      OF THE SYSTEM STATE VARIABLES                            011527
C      THESE MUST BE COMPATABLE WITH THAT DEFINED IN CONTROL   011528
C      IN CONTROL THE DEFAULT VALUES ARE:                      011529
C      NXSS = 1                                                  011530
C      NBTQ = 0                                                  011531
C      NAXX = NXSS + NBTQ                                       011532
C      DEFAULT EQUATION FIRST UNCONSTRAINED BETA ANGLE         011533
C      YDT(NEQ+1) = Y(LCCU(2*NB+1))                             011534
C
C      SET OF AUXILIARY EQUATIONS FOR OBTAINING TRANSFER FUNCTIONS 011535
C      BETWEEN PHYSICALLY REALIZABLE SENSOR SIGNALS AND APPLIED 011536
C      TORQUES                                                  011537
C      EXAMPLE:                                                 011538
C      NAXX = 6                                                  011539
C      LDEL = LCCU(2*NB+2) - 1                                  011540
C      ACCN = 57.295800                                         011541
C      YDT(NEQ+1) = ACCN*RCL(3,2,1)/RCL(3,3,1)                  011542
C      YDT(NEQ+2) = -ACCN*RCL(3,1,1)/RCL(3,3,1)                 011543
C      YDT(NEQ+3) = ACCN*RCL(2,1,1)/RCL(2,2,1)                  011544
C      YDT(NEQ+4) = Y(LDEL+2)                                    011545
C      YDT(NEQ+5) = Y(LDEL+4)                                    011546
C      YDT(NEQ+6) = Y(LDEL+6)                                    011547
C      RETURN                                                    011548
C      END                                                        011549
C
C      SUBROUTINE GMISC (N,LE,LC,V2)                             011550
C      IMPLICIT REAL*8 (A-H,O-Z)                                011551
C      DIMENSION V2(1)                                          011552
C
C      DEBUG # 111                                              011553
C
C      IMPLICIT REAL*8 (A-H,O-Z)                                011554
C      DIMENSION V2(1)                                          011555

```

```

C
      COMMON /MAXIMUM/
*      NBMAX,NHMAX,NSPMAX,NMWMAX,NMWEDD,NMOJJD,<MO,KY,KU
      COMMON /SPECIF/
*      BETAH(6, 6),BETAHD(6, 6),AMD(2, 5),RH(3,3,30),RS(3,3,30),
*      CH(3,35),DS(3,30),IND(3, 5),NMOW(6, 5),IFTSM(15),
*      NB,NH,NSPT,NCFMO,NDELTA,ITCPUL(2, 6),IRGFLX( 6),IHDATA(7, 6),
*      LOCL(14),LENU(14),NU,NBETA,NLAM,NEQ
      COMMON /VECTOR/
*      Y(250),YDT(250)
C
      DATA IIST / 0 /
C
CCC  USER SUPPLIED SUBROUTINE TO CREAT MISC. CONTRIBUTIONS TO R.H.S.
CCC  INCLUDING THE THERMAL GRADIENT ENVIRONMENT.
C      IF THERMALLY INDUCED MOTION MODEL REQUIRED REPLACE 0.00 BY
C      THE THERMAL POSITION THE PARTICULAR MODE IN THE DO LOOP
C      APPROPRIATE COMPUTATION OF THE THERMAL POSITION MUST BE
C      CARRIED OUT. PREFERABLE IN CNTRL DATA TRANSFERED BY A
C      SPECIAL PURPOSE COMMON BLOCK
      IF (IIST .EQ. 1) GO TO 5
      IIST = 1
C
      E CONTINUE
      LGM1 = LC - 1
      DO 10 I=1,LL
10  V2(I) = Y(LGM1 + I) - 0.0 0
C
      RETURN
      END

```

011556
011557
011558
011559
1611560
1711561
1811562
1911563
011564
2011565
011566
011567
011568
011569
011570
011571
011572
011573
011574
011575
011576
011577
011578
011579
011580
011581
011582
011583
011584
011585

```

      SUBROUTINE DEFINITION
      DEBUG # 112
C
C
C
C
C
C
C      COMMON BLOCK / SUBROUTINE
C      CROSS-REFERENCE
C      MATRIX
C
C      ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz
C      MNOPQRSTUVWXYZABCDEFGHIJKLMnopqrstuvwxyz
C      CENAAOSNITCLSMODESJAUTIU12ORMXSMNMNTWTREMPMCN
C      ESPUTAAONGOFIBUABIVNCARTL  RKCMCESBPDUJKMEETO
C      WRAXITVSEKRLZUNTUOZ ATRTL  KVDUNN RRTKFTIRNSOE
C      OR CAE RLSGEATAGE Y TLE  IIEVGG S AKFAFYOSRP
C      ADT
C      ACOT
C      ASIMLR
C      EABT
C      EAKSLV
C      ECOTOP
C      RTGENR

```

011586
011587
011588
011589
011590
011591
011592
011593
011594
011595
011596
011597
011598
011599
011600
011601
011602
011603
011604
011605
011606
011607
011608
011609

C	ESGENR	*	*			*	*	*		011610		
C	CCNTRL	**		*				*	**	011611		
C	CREA					*		*		011612		
C	CREADC					*		*		011613		
C	CREB					*		*		011614		
C	CREC					*		*		011615		
C	CREE					*		*		011616		
C	CREMUD					*		*		011617		
C	CRET3					*		*		011618		
C	DYNS10	*	*	*		**	*	*	****	011619		
C	DYNS20	*	*	*	*	**	*	*	****	011620		
C	DYNS30						*	*	*	011621		
C	DYNS40			*	*****	*	*	*	***	011622		
C	ENGNCM	*		*		*	*	*	*	011623		
C	EGADD	*	*		*	*		*	*	011624		
C	EXTCR					*		*	*	011625		
C	FINDCT				***			*	*	011626		
C	FINDU	**		*		*	*	*	***	011627		
C	GALSSI		*							011628		
C	GETEMB	**		*	*	*	*	*		011629		
C	GMISC					*		*	*	011630		
C	GRVCRD	**	**	*	*	*	*	*	*	011631		
C	INVINP			*						011632		
C	KHINGE	**				*	*	*		011633		
C	LINEAR			**			*	*	**	011634		
C					ABCDEFGHIJKLMNPPPPSSSTTVV					011635		
C					PHONKGGALMVIDACBDDIKRSSSTTVVWVZAIOHUIERSPPJATEI					011636		
C					UENAAUGNITCLSMODESJAUTIGI20PMXSMNMNTWIKEMPMCN					011637		
C					ESPUTAADNGDFISGABIVRCARTL		PKOMCESBPDUJKMELTU			011638		
C					WKAXITVBERNLZONTUZ	ATKIL	KVOUAN	RRKFTIRVSUE		011639		
C					ORCAE	RLSGEATAGE	Y	IL	LIEMGG	S	AKFAFYGSRP	011640
C	LPLTWR			*			*	***			011641	
C	LPRNT			*				**			011642	
C	LIGRQL			*				**			011643	
C	LITRESP			*	*		*	***			011644	
C	MAIN	**	*				*	*	*	*	011645	
C	MGEN	**	*	*	*		*	*	*	*	011646	
C	MRIGIC						*	***			011647	
C	MSMDC						*	***			011648	
C	MSMCDL						*	***			011649	
C	MULTB					*					011650	
C	NIPLCT				**	*		*			011651	
C	NUMS	*									011652	
C	NYPLOT				**	*		*			011653	
C	PAGEHD				**						011654	
C	PLOTWR			*		*	*	*	***		011655	
C	PLTCAR				**	*		*			011656	
C	PRNTOU	**		*		***		*	*		011657	
C	QRDRVR			*		*					011658	
C	RKACAM		*					*	*	***	011659	
C	RLECCUS					*		*			011660	
C	RPLPLOT				**	*		*			011661	
C	RETCR	*	*	*			*	*	*		011662	
C	RGICG		*					*	*		011663	
C	RECTR						*		*		011664	
C	SFRLOZ		*	*		*		*			011665	
C	SHAFFT					*		*	*		011666	
C	SPLCT				**	*		*			011667	
C	START			*	**	*					011668	
C	TFPLY								*		011669	

```

C 1FIYPE * * * * * 011670
C TORQUE * * * * * 011671
C WTAPE ** 011672
C YDOT ** * * * * * 011673
C ABCDDGGHIIJKLLLLLLLLLLLLLLLLLLLLMMNNPPPPQSSSTTVV 011674
C MHONRGALNVIDACBDDIARSSTTVVWZAIQHUIRSPPJALEI 011675
C LBNAADSNITCLSMODESJACTTII120RMXSMNMNTWTREMPMCN 011676
C ESPUTAADNGOFIBGABIVRCARTL RKOMCESBPDUJKMEETD 011677
C WRAXITVSEARNLZONTUZ ATRL KVDUNN RRTREFIRNSUE 011678
C DR CAL RLSGEATAGE Y TIE I1EMOG S AKFAFYUSRP 011679
C 011680
C 011681
C 011682
C 011683
C 011684
C ***** 011685
C 011686
C COMMON /AMUBW/ 011687
C USED IN: 011688
C DYN20 GETBMB PRNTOU 011689
C FINOU MGLN 011690
C YDOT GRVGRD 011691
C PARAMETERS: 011692
C AMU = INERTIA MATRIX FOR BODY N, AT T=0 COMPUTED VIA DYN20 STORED 011693
C ON NTAPE1, THEREAFTER UPDATED IN MGEN 011694
C EW = COEFFICIENT MATRIX OF EQUATION 11-84, VOL 1 011695
C SUBPARTITION OF WHICH IS INVERTED, COMPUTED IN FINOU 011696
C AREA ALSO USED IN YDOT AND GETBMB FOR WORK ASSOCIATED WITH 011697
C COMPUTATION OF LAGRANGE MULTIPLIERS 011698
C 011699
C ***** 011700
C 011701
C COMMON /BHBSRD/ 011702
C USED IN: 011703
C FINOU GETBMB ENGMCM 011704
C YDOT MGEN KHINGE 011705
C ROTDH GRVGRD CNTRL 011706
C BHGENR BDOTUP PRNTOU 011707
C BSGENR TORQUE EQADD 011708
C PARAMETERS: 011709
C BH = ARRAYS OF ELEMENTS WHICH DEFINE HINGE POINT KINEMATICS 011710
C COMPUTED IN BHGENR, SEE PAGE 36, VOL 1 011711
C ES = ARRAYS OF ELEMENTS WHICH DEFINE SENSOR POINT KINEMATICS 011712
C COMPUTED IN BSGENR, SEE PAGE 17, VOL 11 011713
C RCL = ROTATION TRANSFORMATIONS RELATING THE BODY AXIS SYSTEM 011714
C TO THE INERTIAL REFERENCE, COMPUTED IN ROTDH, SEE PAGE 18, 011715
C VOL 2 011716
C DOL = VECTOR COMPONENTS FROM INERTIAL REFERENCE TO THE BODY FIXED 011717
C AXIS SYSTEM, COMPUTED IN ROTDH, SEE PAGE 18, VOL 11 011718
C 011719
C ***** 011720
C 011721
C COMMON /CONPAR/ 011722
C USED IN: 011723
C DYN20 CNTRL 011724
C DYN20 (ANY OTHERS REQUIRED BY USER) 011725
C KHINGE 011726
C PARAMETERS: 011727
C CNIDTA = INPUT IN DYN20, IN DYN20 THE FIRST NDELTA ELEMENTS ARE 011728
C THE INITIAL CONDITIONS FOR THE NDELTA DIFFERENTIAL 011729

```

```

C          EQUATIONS DEFINED IN CONTROL. THE REST ARE USER DEFINED.      011730
C          THE USER MAY USE THIS AREA TO INPUT PARAMETERS NEEDED        011731
C          IN ANY USER DEFINED SUBROUTINE                                011732
C                                                                           011733
C*****                                                                    011734
C                                                                           011735
C          COMMON /LNAUX/                                                011736
C          USED IN:                                                       011737
C          DYNSEC                                                         011738
C          EQADD                                                         011739
C          PARAMETERS:                                                    011740
C          NAUX = NUMBER OF AUXILIARY EQUATIONS USED TO DEFINE THE SENSOR 011741
C          AND THROU SIGNALS REQUIRED FOR THE STABILITY ANALYSIS          011742
C          NAUX = NASS+NBTO (IN EQADD)                                     011743
C                                                                           011744
C*****                                                                    011745
C                                                                           011746
C          COMMON /LKRATIO/                                              011747
C          USED IN:                                                       011748
C          MAIN                                                           011749
C          GAUSS1                                                         011750
C          NUMS                                                            011751
C          PARAMETERS:                                                    011752
C          IFL1 = INTEGER FLAG, DEFINES IF CALL TO GAUSS1 IS FROM NUMS.   011753
C          IF SO IFL1=1. INITIALIZE TO ZERO IN MAIN                      011754
C          IFL2 = INTEGER FLAG, IF CALL IS FROM NUMS AND IFL2 SET =1 IN  011755
C          GAUSS1 A ZERO DENOMATOR GAIN CONDITION ENCOUNTERED          011756
C          DRVBC = PIVOT ELEMENTS USED IN GAUSS1 TO COMPUTE A**(-1) THESE 011757
C          IN TURN TO BE USED TO GET DET(A)                             011758
C                                                                           011759
C*****                                                                    011760
C                                                                           011761
C          COMMON /GDATA/                                                011762
C          USED IN:                                                       011763
C          MAIN                                                           011764
C          DYNSEC                                                         011765
C          GRVGRD                                                         011766
C          PARAMETERS:                                                    011767
C          GAMG1 = INPUT DATA FROM DYNSEC COMPONENTS OF GRAVITY VECTOR IN 011768
C          INERTIAL COORDINATES. SEE SUBROUTINE DEF3 REDEFINED            011769
C          IN DYNSEC TO BE COMPONENTS OF UNIT VECTOR IN DIRECTION        011770
C          OF GRAVITY FIELD, EQUIVALENCE(GAMG1(I),WV(I)) IN DYNSEC        011771
C          (FROM GRAVITY SOURCE TO VICINITY OF MODEL CENTER OF MASS)    011772
C          CMAG = MAGNITUDE OF LOCAL GRAVITATIONAL ACCELERATION, COMPUTED 011773
C          IN DYNSEC, SEE PAGE 46, VOL I (MAG. OF INPUT GAMG1)          011774
C          FCMAG = MAGNITUDE OF VECTOR FROM GRAVITY SOURCE TO VICINITY OF 011775
C          SPACECRAFT, INPUT DATA FROM DYNSEC                           011776
C                                                                           011777
C*****                                                                    011778
C                                                                           011779
C          COMMON /GCSAVE/                                                011780
C          USED IN:                                                       011781
C          MGEN                                                           011782
C          GRVGRD                                                         011783
C          TORQUE                                                         011784
C          PARAMETERS:                                                    011785
C          CGS = GRAVITY GRADIENT EFFECTS ON THE ELASTIC COORDINATES OF EACH 011786
C          BODY, COMPUTED IN MGEN SEE PAGE 52,VOL I                     011787
C                                                                           011788
C*****                                                                    011789

```


C		C11790
C	COMMON /HANDS/	C11791
C	USED IN:	C11792
C	DYN20 BSGENR BDDTOP	C11793
C	YDOT ROTDS	C11794
C	ROTOR BSGENR	C11795
C	PARAMETERS:	C11796
C	HATH = MODAL DISPLACEMENTS AT HINGE POINTS	C11797
C	SICH = MODAL ROTATIONS AT HINGE POINTS	C11798
C	HATS = MODAL DISPLACEMENTS AT SENSOR POINTS	C11799
C	SICS = MODAL ROTATIONS AT SENSOR POINTS	C11800
C		C11801
C	*****	C11802
C		C11803
C	COMMON /ILINER/	C11804
C	USED IN:	C11805
C	MAIN	C11806
C	DYN20	C11807
C	DYN20	C11808
C	YDOT	C11809
C	PARAMETERS:	C11810
C		C11811
C	IFLNER = 1 PERFORM LINEAR TIME OR FREQUENCY RESPONSE ANALYSIS	C11812
C	= 0 PERFORM NON-LINEAR TIME RESPONSE ANALYSIS	C11813
C	THIS INTEGER IS SET IN DYN20 BY DATA INPUT PARAMETER	C11814
C	IPDATA(3), SEE SUBROUTINE DEFS	C11815
C		C11816
C	*****	C11817
C		C11818
C	COMMON /INTGR/	C11819
C	USED IN:	C11820
C	DYN20 GRVGRD	C11821
C	YDOT TORQUE	C11822
C	MGEN ENGMOM	C11823
C	PARAMETERS:	C11824
C	AM = UNDEFORMED INERTIA MATRIX (M SUB 0 IN EQ 11-87)	C11825
C	ACCF = ALPHA COEFFICIENTS OF EQ 11-88 USED TO DEFINE BODY'S	C11826
C	STATIC MASS MOMENTS LINEAR DEPENDENCE ON DEFORMATION	C11827
C	ECCF = 6 COEFFICIENTS OF EQ 11-88 USED TO DEFINE BODY INERTIA	C11828
C	TENSOR'S LINEAR DEPENDENCE ON DEFORMATION	C11829
C	COF11 = C SUB 11 COEFFICIENTS OF EQ 11-89, INERTIA TENSOR QUAD DEP	C11830
C	COF22 = C SUB 22 COEFFICIENTS OF EQ 11-89, INERTIA TENSOR QUAD DEP	C11831
C	COF33 = C SUB 33 COEFFICIENTS OF EQ 11-89, INERTIA TENSOR QUAD DEP	C11832
C	AK = MODAL STIFFNESS MATRIX	C11833
C	COF12 = C SUB 12 COEFFICIENTS OF EQ 11-89, INERTIA TENSOR QUAD DEP	C11834
C	COF13 = C SUB 13 COEFFICIENTS OF EQ 11-89, INERTIA TENSOR QUAD DEP	C11835
C	COF23 = C SUB 23 COEFFICIENTS OF EQ 11-89, INERTIA TENSOR QUAD DEP	C11836
C	AD = MODAL DAMPING MATRIX	C11837
C	COFXY = C SUB XY COEFFICIENTS OF EQ 11-89	C11838
C	COFXZ = C SUB XZ COEFFICIENTS OF EQ 11-89	C11839
C	COFYZ = C SUB ZY COEFFICIENTS OF EQ 11-89	C11840
C		C11841
C	*****	C11842
C		C11843
C	COMMON /IVCONS/	C11844
C	USED IN:	C11845
C	YDOT	C11846
C	GETUMB	C11847
C	PARAMETERS:	C11848
C	IV = INTEGER ARRAY SET UP IN GETUMB, USED TO PERFORM THE SELECTIVE	C11849

```

C          ROW MULTIPLICATIONS REQUIRED IN GET8MB, PICKS OUT CONSTRAINED 011850
C          AND UNCONSTRAINED COORDINATES AT EACH HINGE POINT. 011851
C          PROVIDES THE NUMBERING SEQUENCE FOR THE LAGRANGE MULTIPLIERS 011852
C          EVALUATED IN YDOT 011853
C 011854
C*****011855
C 011856
C      COMMON /JILFLG/ 011857
C      USED IN: 011858
C          YDOT 011859
C          RKADAM 011860
C      PARAMETERS: 011861
C      JIL = 1,2,3 OR 4 DEFINE CYCLE NUMBER IN FOUR STEP RUNGE KUTTA 011862
C      INTEGRATION ROUTINE. USED IN YDOT TO SET FLAG WHICH SIGNALS 011863
C      WHETHER OR NOT A NEW SET OF INDEPENDENT COORDINATES SHOULD 011864
C      BE EVALUATED IN FINDU 011865
C 011866
C*****011867
C 011868
C      COMMON /KOSIZE/ 011869
C      USED IN: 011870
C          DYN540      LTRCOL 011871
C          SFRB02      LTRFSP 011872
C      PARAMETERS: 011873
C      KR  = DIMENSION OF ARRAYS IN /LIJV/,/LRARAY/,/LV1/,/LV2/,/LWORK1/ 011874
C      KV1 = DIMENSION OF ARRAYS IN /LIJV/,/LRARAY/,/LV1/,/LV2/,/LWORK1/ 011875
C      KRT = DIMENSION OF ARRAY R  IN /LRGCT/ 011876
C      KRX = DIMENSION OF ARRAY RX IN /LRGCT/ 011877
C      KV2 = DIMENSION OF ARRAY R  IN /LRGCT/ 011878
C      KRX = DIMENSION OF ARRAY RX IN /LRGCT/ 011879
C 011880
C*****011881
C 011882
C      COMMON /LAMDDA/ 011883
C      USED IN: 011884
C          YDOT 011885
C          PRINTU 011886
C          PLOTWR 011887
C      PARAMETERS: 011888
C      ALAM = IN YDOT EQUIVALENCE (ALAM(1),V(1)) AREA USED TO COMPUTE 011889
C      LAGRANGE MULTIPLIERS, UPON EXIT FROM YDOT, ALAM ARRAY 011890
C      CONTAINS THE NLAM INTERCONNECTION CONSTRAINT FORCES 011891
C      (LAMBDAS) 011892
C 011893
C*****011894
C 011895
C      COMMON /LQDATA/ 011896
C      USED IN: 011897
C          DYN540 011898
C      PARAMETERS: 011899
C      FMIN = FREQUENCY SWEEP LOWER LIMIT (DYN540) 011900
C      FMAX = FREQUENCY SWEEP UPPER LIMIT (DYN540) 011901
C      EBMIN = MINIMUM DB AMPLITUDE FOR BODE, NICHOLS PLOTS (DYN540) 011902
C      DBMAX = MAXIMUM DB AMPLITUDE FOR BODE, NICHOLS PLOTS (DYN540) 011903
C 011904
C*****011905
C 011906
C      COMMON /LCOUNT/ 011907
C      USED IN: 011908
C          DYN540 011909

```

```

C      PARAMETERS: 011910
C      NNR = NUMBER OF REAL AND ZERO ROOTS IN NUMERATOR (DCQRT) 011911
C      ICN = NUMBER OF COMPLEX PAIRS IN NUMERATOR (DCQRT) 011912
C      NNZ = NUMBER OF ZEROS IN NUMERATOR (DCQRT) 011913
C      NOR = NUMBER OF REAL AND ZERO ROOTS IN DENOMINATOR (DCQRT) 011914
C      ICD = NUMBER OF COMPLEX PAIRS IN DENOMINATOR (DCQRT) 011915
C      NDZ = NUMBER OF ZEROS IN DENOMINATOR (DCQRT) 011916
C 011917
C 011918
C 011919
C 011920
C***** 011921
C 011922
C      COMMON /LDEBUG/ 011923
C      USED IN: 011924
C      (ALL ROUTINES HAVING DEBUG PRINTING CODED) 011925
C      PARAMETERS: 011926
C      DEBUG = LOGICAL ARRAY SKIP DEBUG PRINT IN SUBROUTINE (DEBUG # K) 011927
C      IF DEBUG(K) = .FALSE., PRINT IF DEBUG(K) = .TRUE. 011928
C 011929
C***** 011930
C 011931
C      COMMON /LOSIZE/ 011932
C      USED IN: 011933
C      LINEAR FINDT LPLTWR EQADD 011934
C      DYN540 Tftype LPRNT 011935
C      ASIMLR LTRESP CONTRL 011936
C      PARAMETERS: 011937
C      NXSS = NUMBER OF PLANT SENSOR SIGNALS, USER DEFINED IN 011938
C      CONTRL, DEFAULT IS 1 (FOR STABILITY ANALYSIS) 011939
C      NBTQ = NUMBER OF TORQUE SIGNALS, USER DEFINED IN 011940
C      IN CONTRL, DEFAULT IS 0 HOWEVER FOR STABILITY ANALYSIS 011941
C      NBTQ MUST BE AT LEAST ONE, NBTQ = NB IN SOME ROUTINES 011942
C      NDLTA = NUMBER OF DIFFERENTIAL EQUATIONS DEFINED IN CONTRL 011943
C      EQUAL TO INPUT PARAMETER NDLTA IN CONTRL 011944
C      NX = NUMBER OF INDEPENDENT COORDINATES IN THE STATE VECTOR Y 011945
C      COMPUTED IN LINEAR 011946
C      NJG = NX + NUMBER OF AUXILIARY EQUATIONS, COMPUTED IN LINEAR 011947
C      NY2 = NX - NDLTA - NXSS, COMPUTED IN DYN540 011948
C      ND2 = NDLTA - NBTQ, COMPUTED IN DYN540 (NBTQ=NB) 011949
C      NY = UNUSED 011950
C 011951
C***** 011952
C 011953
C      COMMON /LIJY/ 011954
C      USED IN: 011955
C      DYN540 011956
C      FINDT 011957
C      PARAMETERS: 011958
C      IV = INTEGER WORK VECTOR PASSED TO ASIMLR VIA DYN540, USED THERE 011959
C      FOR REORDERING R**(-1)*H*. EXIT FINDT WITH IV BEING THE 011960
C      TRANSFORMED STATE VECTOR CORRELATION ARRAY 011961
C      JV = INTEGER WORK AREA USED IN FINDT 011962
C 011963
C***** 011964
C 011965
C      COMMON /LKRAY/ 011966
C      USED IN: 011967
C      DYN540 011968
C      PARAMETERS: 011969

```

```

C      FERN = TIME CONSTANTS ASSOCIATED WITH ALL REAL ROOTS OF NUMERATOR      011970
C      FENC = FREQUENCIES OMEGA AND DAMPING ZETA'S ASSOCIATED WITH ALL      011971
C      COMPLEX ROOTS OF THE NUMERATOR                                         011972
C      FROM:                                                                    011973
C      S**2 + 2*ZETA*OMEGA*S + OMEGA**2 = 0                                   011974
C      FEND = TIME CONSTANTS ASSOCIATED WITH ALL REAL ROOTS OF DENOMIATOR    011975
C      FDEC = FREQUENCIES OMEGA AND DAMPING ZETA'S ASSOCIATED WITH ALL      011976
C      COMPLEX ROOTS OF THE DENOMINATOR                                       011977
C      ** NCIL ** NUMERATOR/DENOMINATOR = TRANSFER FUNCTION REQUESTED       011978
C                                                                              011979
C*****                                                                    011980
C      COMMON /LRROOT/                                                         011981
C      USED IN:                                                                011982
C      DYN540                                                                    011983
C      PARAMETERS:                                                             011984
C      R = ROOTS ARRAY COMPUTED IN TFF FOR PARTICULAR TRANSFER FUNCTION      011985
C      ROOTS COMMON TO DENOMINATOR AND NUMERATOR CANCELLED IN CANCEL        011986
C      RX = ARRAY USED TO STORE ALL COEFFICIENTS OF THE NUMERATOR AND        011987
C      DENOMINATOR POLYNOMIALS, ROOTS IN R USED TO COMPUTE THESE            011988
C      COEFFICIENTS IN RIOP                                                  011989
C                                                                              011990
C*****                                                                    011991
C      COMMON /LSTART/                                                         011992
C      USED IN:                                                                011993
C      START      SPLIT      REPLOT                                           011994
C      PAGEHD     NIPLT      PLTCAR                                           011995
C      WTAPE      NYPLT                                           011996
C      PARAMETERS:                                                             011997
C      IRUNNL = RUN NUMBER, INPUT IN START                                     011998
C      IDATE = DATA OF RUN FROM BINARY ROUTINE                               011999
C      NPAGE = PAGE COUNTER UPDATED IN PAGEHD                                 012000
C                                                                              012001
C*****                                                                    012002
C      COMMON /LSTRTI/                                                         012003
C      USED IN:                                                                012004
C      START      SPLIT      REPLOT                                           012005
C      PAGEHD     NIPLT      PLTCAR                                           012006
C      WTAPE      NYPLT                                           012007
C      PARAMETERS:                                                             012008
C      UNAME = USERS NAME, INPUT START                                       012009
C      TITLE1 = FIRST TITLE EACH PAGE, INPUT START                          012010
C      TITLE2 = SECOND TITLE EACH PAGE, INPUT START                         012011
C                                                                              012012
C*****                                                                    012013
C      COMMON /LTITLE/                                                         012014
C      USED IN:                                                                012015
C      DYN540                                                                    012016
C      PARAMETERS:                                                             012017
C      TITLE = INPUT FORMAT 20A4, PUT ON ALL FREQUENCY DOMAIN OUTPUT DATA 012018
C      PRINT-OUT AND PLOTS                                                  012019
C                                                                              012020
C*****                                                                    012021
C      COMMON /LTUL/                                                           012022
C      USED IN:                                                                012023
C      DYN540                                                                    012024
C                                                                              012025
C*****                                                                    012026
C      COMMON /LTUL/                                                           012027
C      USED IN:                                                                012028
C      DYN540                                                                    012029

```

C	PARAMETERS:	012030
C	TOLN = TOLD = TOLERANCE PARAMETERS ALL NUMERATOR AND DENOMINATOR	012031
C	ROOT LESS THAN TOLN.TOLD ARE SET TO ZERO IN CALL TO	012032
C	SIFT VIA DYN540	012033
C		012034
C	*****	012035
C	COMMON /LV1/	012036
C	USED IN:	012037
C	DYN540 RLOCUS	012038
C	SFREQ2 LTRESP	012039
C	PARAMETERS:	012040
C	V1,V2,V3 = TEMPORARY WORK AREAS	012041
C		012042
C	*****	012043
C	COMMON /LV2/	012044
C	USED IN:	012045
C	DYN540	012046
C	PARAMETERS:	012047
C	XV1 = WORK AREA USED IN DYN540 FOR FREQUENCY RESPONSE ANALYSIS	012048
C	XV2 = WORK AREA USED IN DYN540 FOR FREQUENCY RESPONSE ANALYSIS	012049
C	XV3 = WORK AREA USED IN DYN540 FOR FREQUENCY RESPONSE ANALYSIS	012050
C	XV4 = WORK AREA USED IN DYN540 FOR FREQUENCY RESPONSE ANALYSIS	012051
C		012052
C	*****	012053
C	COMMON /LWORK1/	012054
C	USED IN:	012055
C	DYN540	012056
C	QRQRVR	012057
C	LTRESP	012058
C	PARAMETERS:	012059
C	W1 = USED AS WORK AREAS	012060
C	W2 = USED AS WORK AREAS	012061
C		012062
C	*****	012063
C	COMMON /LWRKV1/	012064
C	USED IN:	012065
C	BABT	012066
C	MULTB	012067
C	PARAMETERS:	012068
C	W = WORK AREA	012069
C		012070
C	*****	012071
C	COMMON /LZMODE/	012072
C	USED IN:	012073
C	START NIPLT	012074
C	PLTCAR NYPLT	012075
C	SPLT RLPLT	012076
C	PARAMETERS:	012077
C	AMODE = ARRAY SET UP BY SC4020 ROUTINE MODESG	012078
C		012079
C	*****	012080
C	COMMON /MAXMUM/	012081
C	USED IN:	012082
C	MAIN YOUT GRVGRD KHINGE PRNTOU	012083
C		012084
C		012085
C		012086
C		012087
C		012088
C		012089

```

C          DYN510    BMSGNR    BOUTOP    EXTOR    012090
C          MSMDDC    BSGENR    TORQUE    STAFFT    012091
C          DYN520    GETBMB    ENGMCM    EQAJJ    012092
C          FINDU    MUEN    PLOTWR    GMISC    012093
C      PARAMETERS:    012094
C      NBMAX = MAXIMUM NUMBER OF BODIES    012095
C      NHMAX = MAXIMUM NUMBER OF HINGES    012096
C      NSPMAX = MAXIMUM NUMBER OF SENSOR POINTS    012097
C      NMWMAX = MAXIMUM NUMBER OF MOMENTUM WHEELS    012098
C      NMWBCD = MAXIMUM NUMBER OF MOMENTUM WHEELS PER BODY    012099
C      NMCHCD = MAXIMUM NUMBER OF FLEXIBLE BODY MODES PER BODY    012100
C      KMU = MAXIMUM NUMBER OF VELOCITY VARIABLES PER BODY (U'S)    012101
C      KY = MAXIMUM NUMBER OF PARAMETERS IN THE STATE VECTOR    012102
C      KU = MAXIMUM NUMBER OF VELOCITY VARIABLES FOR SYSTEM    012103
C      (ALL OF THESE PARAMETERS ARE SET IN MAIN, TO CHANGE ANY OR    012104
C      ALL OF THEM THE REDIMENSION PROGRAM MUST BE USED WHICH    012105
C      WILL REDIMENSION THE APPROPRIATE PARAMETERS IN THE PROGRAM    012106
C      REDIMENSIONED PROGRAM MUST THEN BE RECOMPILED)    012107
C    012108
C*****    012109
C    012110
C      COMMON /MISCNO/    012111
C      USED IN:    012112
C          MAIN    PRNTOU    012113
C          DYN510    LTRSP    012114
C          DYN520    DYN540    012115
C      PARAMETERS:    012116
C      NOPRNT = INPUT PARAMETER IPDATA(1) READ IN DYN510, PRINT OUTPUT    012117
C              DATA EVERY NOPRNT MULTIPLES OF INTEGRATION STEP    012118
C      NOPLCT = INPUT PARAMETER IPDATA(2) READ IN DYN510, PLOT CONTROL    012119
C              FOR TIME RESPONSE PLOT EVERY NOPLCT MULTIPLES OF    012120
C              INTEGRATION STEP    012121
C    012122
C    012123
C    012124
C*****    012125
C    012126
C      COMMON /MEMENG/    012127
C      USED IN:    012128
C          YDOT    PRNTOU    012129
C          TORQUE    PLOTWR    012130
C          ENGMOM    KHINGE    012131
C      PARAMETERS:    012132
C      P = ORDINARY MOMENTA COMPUTED IN YDOT, SEE EQUATION    012133
C          11-49, VOL 1    012134
C      FMCM = CONTRIBUTION TO TOTAL ANGULAR AND LINEAR MOMENTUM OF    012135
C          EACH BODY, COMPUTED IN ENGMOM    012136
C      FICT = TOTAL SYSTEM ANGULAR MOMENTUM VECTOR, COMPUTED IN ENGMCM    012137
C      TOTAL = TOTAL SYSTEM LINEAR MOMENTUM VECTOR, COMPUTED IN ENGMOM    012138
C      ENKGL = CONTRIBUTION TO TOTAL KINETIC ENERGY OF EACH BODY,    012139
C              COMPUTED IN ENGMOM    012140
C      ENSPE = CONTRIBUTION TO TOTAL POTENTIAL ENERGY OF EACH BODY,    012141
C              COMPUTED IN ENGMOM    012142
C      TOTKE = TOTAL KINETIC ENERGY    COMPUTED IN ENGMOM    012143
C      TOTPE = TOTAL POTENTIAL ENERGY,    COMPUTED IN ENGMOM    012144
C      TOTENG = TOTAL ENERGY (T+V)    ,    COMPUTED IN ENGMOM    012145
C      ATOTL = TOTAL ANGULAR MOMENTUM    ,    COMPUTED IN ENGMOM    012146
C      ATOTL = TOTAL LINEAR MOMENTUM    ,    COMPUTED IN ENGMOM    012147
C    012148
C*****    012149

```

C		012150
C	COMMON /NHNS/	012151
C	USED IN:	012152
C	DYN510 MRIGID	012153
C	MSMODL	012154
C	MSMODC	012155
C	PARAMETERS:	012156
C	NHPOI = NUMBER OF HINGES ON EACH BODY, COMPUTED IN DYN510	012157
C	NSPOI = NUMBER OF SENSOR POINTS ON EACH BODY, COMPUTED IN DYN510	012158
C		012159
C	*****	012160
C		012161
C	COMMON /NUMBRS/	012162
C	USED IN:	012163
C	MAIN CREA FINDU GETBMB ENGMCM	012164
C	MSMODC CRLB YDOT MGEN CREMUO	012165
C	MSMODL CREC CREE GRVGRD	012166
C	CRETS RUTR BHGENR BOOTOP	012167
C	CREADD DYN520 BSGENR TORQUE	012168
C	PARAMETERS:	012169
C	ZRO = 0.00 (ALL SET IN MAIN)	012170
C	CNE = 1.00	012171
C	TWC = 2.00	012172
C	TRES = 3.00	012173
C		012174
C	*****	012175
C		012176
C	COMMON /PINPR/	012177
C	USED IN:	012178
C	RUTDH	012179
C	BHGENR	012180
C	BOOTOP	012181
C	PARAMETERS:	012182
C	PIN = ROTATION TRANSFORMATION MATRIX RELATING BODY FIXED ANGULAR	012183
C	RATES TO EULER RATES BETWEEN P AND Q FRAME AT EACH HINGE,	012184
C	COMPUTED IN RUTDH	012185
C	RP2 = COMPUTED IN RUTR VIA CALL FROM RUTDH	012186
C	RP3 = COMPUTED IN RUTR VIA CALL FROM RUTDH	012187
C	(BOTH RP2,RP3 USED IN BOOTOP TO OBTAIN $\partial/\partial t(P1(INVERSE))$)	012188
C		012189
C	*****	012190
C		012191
C	COMMON /PLTDTA/	012192
C	USED IN:	012193
C	MAIN DYN530	012194
C	DYN520 DYN540	012195
C	PLUTWR LPLTWR	012196
C	PARAMETERS:	012197
C	NRPLCT = BOTH ARE INTEGERS COMPUTED IN PLTWR, LPLTWR AND DYN530	012198
C	NCPLCT THEY ARE USE IN DYN530 FOR CREATION OF PLOTS	012199
C		012200
C	*****	012201
C		012202
C	COMMON /PRWORK/	012203
C	USED IN:	012204
C	LINLAR	012205
C	RKADAM	012206
C	PARAMETERS:	012207
C	PR = DUAL PURPOSE WORK AREA, TEMPORARY STORAGE AREA FOR ADAMS P/C	012208
C	INTEGRATION ALGORITHM IN RKADAM, TEMPORARY STORAGE AREA IN	012209

```

C          LINEAR FOR OBTAINING PARTIAL DERIVATIVE MATRIX          012210
C                                                                    012211
C ***** 012212
C                                                                    012213
C                                                                    012214
C      COMMON /PSTUFF/                                             012215
C      USED IN:                                                    012216
C          SFREQ2      RIPLDT      DYN540                          012217
C          RLOCUS      NYPLDT                                           012218
C          SPLOT      RIPLDT                                           012219
C      PARAMETERS:                                                 012220
C      KSAVE = TOTAL NUMBER OF POINTS SAVED FOR THE PLOTTING OF THE 012221
C      FREQUENCY RESPONSE DATA
C      SAVED = (IN SFREQ2) THE ARRAY OF FREQUENCIES AT WHICH THE TRANSFER 012222
C      FUNCTION IS EVALUATED THAT IS ALL OMEGA FOR WHICH             012223
C       $TF(1 \times OMEGA) = ALPHA + I \times BETA$                                 012224
C      IS COMPUTED                                                  012225
C      (IN RLOCUS) EQUIVALENCE (SAVED(1),KSAVE(1)) THE ARRAY        012226
C      CONTAINS THE REAL COORDINATES OF THE POINTS FOUND TO LIE     012227
C      ALONG THE ROOT LOCUS                                         012228
C      SAVED = (IN RLOCUS) EQUIVALENCE (SAVED(1),YSAVE(1)) THE ARRAY 012229
C      CONTAINS THE IMAGINARY COORDINATES OF THE POINTS FOUND      012230
C      TO LIE ALONG THE ROOT LOCUS                                  012231
C      (IN SFREQ2)                                                  012232
C          AR = DSQRT(ALPHA**2 + BETA**2)                             012233
C          OPHI = DATAN2(BETA,ALPHA)*57.2956                          012234
C          0 < OPHI < 360.0                                         012235
C          DBELL = 5.0688561*LOG(AR)  WHERE 5.065 = 20*LOG(E)       012236
C      SAVED = DBELL FOR EACH OMEGA                                  012237
C      SAVED = OPHI FOR EACH OMEGA                                  012238
C      SAVED = AR FOR EACH OMEGA                                    012239
C                                                                    012240
C                                                                    012241
C                                                                    012242
C ***** 012243
C                                                                    012244
C      COMMON /CPHNTA/                                             012245
C      USED IN:                                                    012246
C          DYN520                                                  012247
C          RKACAM                                                  012248
C      PARAMETERS:                                                 012249
C      CRK = INITIALIZED IN DYN520 TO 0.00, TEMPORARY STORAGE AREA USED 012250
C      BY NUMERICAL INTEGRATION ROUTINE                             012251
C      PRK = CONSTANTS REQUIRED BY INTEGRATION ALGORITHM, SET IN DYN520 012252
C      AT = COUNTER UPDATE AT END OF EACH INTEGRATION STEP        012253
C                                                                    012254
C ***** 012255
C                                                                    012256
C      COMMON /SPECIF/                                             012257
C      USED IN:                                                    012258
C          MAIN      FINCU      MGEN      PLUTWK      SHAFFT          012259
C          DYN510      YOUT      GRVGRD      ASIMLR      EQADD          012260
C          MSMODC      ENGLENR      EOOTQP      FINJI      GMISC          012261
C          MSMODL      ROTDS      TORQUE      KHINGE      ROTDH          012262
C          MKIGIO      GSGLENR      ENGMCM      CNTRL          012263
C          DYN520      GETUMB      PRNTOU      EXTOR          012264
C      PARAMETERS:                                                 012265
C      BETAH = INPUTTED IN DYN510, CENTIGUOUS HINGE FRAME ORIENTATION 012266
C      INITIAL CONDITIONS. UPDATED RELATIVE ORIENTATION DATA      012267
C      STRIPPED FROM STATE VECTOR Y IN ROTDH. OUTPUTTED             012268
C      IN PRNTOU. SEE PAGE 13,VOL II AND SUBROUTINE DEF3            012269

```


C	BETAFD	= INPUTTED IN DYN510, CONTIGUOUS HINGE FRAME RELATIVE RATE	012270
C		INITIAL CONDITIONS, UPDATED RELATIVE RATE DATA COMPUTED	012271
C		IN YDOT STORED IN YOT AND BETAHD ARRAYS, SEE PAGES	012272
C		18, VOL II AND SUBROUTINE DEF3	012273
C	AMC	= INPUTTED IN DYN510, MOMENTUM WHEEL SPIN RATE AND SPIN	012274
C		INERTIA INITIAL CONDITIONS, VALUES NOT UPDATED, SEE	012275
C		SUBROUTINE DEF3	012276
C	RH	= INTERMEDIATE ROTATION TRANSFORMATIONS USED TO COMPUTE	012277
C		RELATIVE ORIENTATION OF BODY FIXED REFERENCE FRAMES	012278
C		COMPUTED IN ROTCH, SEE COMMENT CARDS THEREIN	012279
C	RS	= ROTATION TRANSFORMATION BETWEEN SENSOR AXIS SYSTEM AND	012280
C		BODY FIXED AXIS SYSTEM, COMPUTED IN ROTDS, SEE ROTDS ALSO	012281
C		SEE PAGE 19, VOL II, INITIAL VALUES COMPUTED IN ROTTR	012282
C		VIA A CALL FROM MRIGID	012283
C	OH	= VECTOR TO POSITION HINGE TRIAD WITH RESPECT TO BODY TRIAD	012284
C		INPUTTED IN MRIGID, IF FLEXIBLE BODY COMPUTED IN MSMODL	012285
C		OR MSMODC, UPDATED IN ROTCH, SEE SUBROUTINE DEF3	012286
C	OS	= VECTOR TO POSITION SENSOR TRIAD WITH RESPECT TO BODY	012287
C		TRIAD INPUTTED IN MRIGID, IF FLEXIBLE BODY COMPUTED IN	012288
C		MSMODL OR MSMODC, UPDATED IN ROTDS, SEE SUBROUTINE DEF3	012289
C	IMO	= BASIC MOMENTUM WHEEL DATA INPUTTED IN DYN510, SEE PAGE	012290
C		SUBROUTINE DEF3, DEFINES SENSOR POINT NUMBER FOR WHEEL,	012291
C		SPIN AXIS NUMBER AND WHETHER IT IS ACTIVE OR CONSTANT	012292
C		SPEED, ALSO SEE SUB. DEF4 OUTPUT DYN510	012293
C	NMCW	= THE MOMENTUM WHEEL/BODY TABLE, SEE DEF4 OUTPUT DYN510	012294
C		DEFINES NUMBER OF WHEELS CN BODY, WHICH ARE VARIABLE	012295
C		SPEED AND RESPECTIVE SENSOR POINT NUMBERS, COMPUTED IN	012296
C		DYN510	012297
C	IFTSW	= SENSOR POINT/BODY CORRELATION ARRAY DEFINES BODY NUMBER	012298
C		CN WHICH EACH SENSOR POINT DEFINED, INPUTTED IN DYN510,	012299
C		SEE SUB DEF3 AND DEF4 INPUT/OUTPUT DYN510	012300
C	NB	= NUMBER OF BODIES IN MODEL, INPUT DYN510, SEE DEF3	012301
C	NH	= NUMBER OF HINGES IN MODEL, INPUT DYN510, SEE DEF3	012302
C	NSPT	= NUMBER OF SENSOR POINTS , INPUT DYN510, SEE DEF3	012303
C	NDFMC	= NUMBER OF MOMENTUM WHEELS, INPUT DYN510, SEE DEF3	012304
C	NDELTA	= NUMBER CONTRL DIFF. EOS. , INPUT DYN510, SEE DEF3	012305
C	ITOPCL	= TOPOLOGY ARRAY, , INPUT DYN510, SEE DEF3 DEFINES	012306
C		THE BODY NUMBERS OF THE CONTIGUOUS BODIES AT EACH HINGE.	012307
C		ALSO DEFINES DIRECTION OF POSITIVE EULER ANGLE ROTATION	012308
C		RGT = A1*A2*A3	012309
C		A1,A2,A3 TRANSFORMATIONS FOR 1-SI,2-ND,3-RD POSITIVE	012310
C		ROTATION, SEE PAGES 4-7, VOL II	012311
C	IRGFLX	= NUMBER OF FLEXIBLE BODY MODES FOR EACH BODY, SEE PAGE	012312
C		4-7, VOL II AND DEF3	012313
C	IHDATA	= HINGE POINT CONSTRAINT DATA, SEE DEF3, INPUT DYN510 PP4-7	012314
C		VOL 2	012315
C	LOCU	= STATE VECTOR LOCATION ARRAY, SEE DEF4, COMPUTED	012316
C		IN DYN510	012317
C	LENU	= STATE VECTOR LENGTH ARRAY, SEE DEF4, COMPUTED IN	012318
C		DYN510	012319
C	NU	= NUMBER OF U'S IN STATE VECTOR, SEE DEF4, PP4-7 OF VOL II	012320
C		COMPUTED IN DYN510	012321
C	NBETA	= NUMBER OF BETA'S IN STATE VECTOR, SEE DEF4 + PP4-7 VOL II	012322
C		COMPUTED IN DYN510	012323
C	NLAM	= NUMBER OF LAMBDA'S (CONSTRAINTS), SEE DEF4 + PP4-7 VOL II	012324
C		COMPUTED IN DYN510	012325
C	NEQ	= NUMBER OF STATE EQUATIONS TO INTEGRATE SEE DEF4	012326
C		COMPUTED IN DYN510	012327
C			012328
C		*****	012329

```

C
C      COMMON /SUMMARY/
C      USED IN:
C          DYN510      MRIGID
C          MSMODL
C          MSMODL
C      PARAMETERS:
C      KSUMRY = ROW DIMENSION OF ASUMRY AND ISUMRY ARRAYS (DYN510)
C      ASUMRY = REAL NUMBER WORK AREA ARRAY
C      ISUMRY = INTEGER NUMBER WORK AREA ARRAY
C
C*****
C      COMMON /TAPEEND/
C      USED IN:
C          MAIN      CREADD      DYN520      DYN540      CREE
C          DYN510      CREA      YDOT      ASIMLR
C          MSMODL      CREC      PLOTWR      LPLTWR
C          MSMODL      CREB      LINEAR      LTRESP
C          CRET3      MRIGID      DYN530      CREMUG
C      PARAMETERS:
C      NTAPE1 = 1 SCRATCH TAPE NUMBER SET IN MAIN
C      NTAPE2 = 2 SCRATCH TAPE NUMBER SET IN MAIN
C      NTAPE3 = 3 SCRATCH TAPE NUMBER SET IN MAIN
C
C*****
C      COMMON /TIMES5/
C      USED IN:
C          DYN510      PRNTOU      LPRNT
C          DYN520      PLOTWR      LTORQL
C          FINDU      DYN540      CENTRL
C          YDOT      LTRESP
C          RKADAM      LPLTWR
C      PARAMETERS:
C      ST = START = TMDATA(1) INPUTTED PARAMETER DYN510 (INITIAL TIME)
C      DT = DELTAT = TMDATA(2) INPUTTED PARAMETER DYN510 (STEP SIZE)
C      T = TIME PARAMETER INCREMENTED IN RKADAM OR LTRESP
C      ET = ENDT = TMDATA(3) INPUTTED PARAMETER DYN510 (END RUN TIME)
C      TMST = 0.0 IN DYN520, TIME ELAPSED FROM START INCREMENTED RKADAM
C           OR LTRESP
C
C*****
C      COMMON /VECTOR/
C      USED IN:
C          DYN520      GRVGRD      DYN540      AJT
C          FINDU      BOUTOP      ASIMLR      AJJ
C          YDOT      TORQUE      TPLY      CONTRL
C          LINEAR      ENGMUM      LTRESP      EXTOR
C          ROTOP      RKADAM      LPLTWR      SHAFHT
C          ROTUS      PRNTOU      LPRNT      EADDJ
C          MGEN      PLOTWR      LTORQL      GMISC
C      PARAMETERS:
C      Y = SYSTEM STATE VECTOR (SEE APPENDIX F, VOL 1 FOR ARRANGEMENT)
C      YD = FIRST TIME DERIVATIVE OF SYSTEM STATE VECTOR, ALL EQUATIONS
C          TO BE INTEGRATED INCLUDED, IN EOADD NAUX AUXILIARY EQUATIONS
C          ADDED FOR LINEAR STABILITY ANALYSIS. (YD1 = YD)
C
C*****

```

C		012390
C	COMMON /VINDEP/	012391
C	USED IN:	012392
C	DYN520 RKADAM	012393
C	FINDU ASIMLR	012394
C	LINEAR FINDT	012395
C	PARAMETERS:	012396
C	INDEP = USED TO DEFINE INDEPENDENT COORDINATES IN THE STATE	012397
C	VECTOR Y. FROM FINDU IF INDEP(I) = 0 COORDINATE I	012398
C	IS DEPENDENT IF = 1 IT IS INDEPENDENT. IN FINDU A OPTIMUM	012399
C	SET OF INDEPENDENT COORDINATES IS EVALUATED AT THE END	012400
C	OF EACH FULL INTEGRATION STEP. IN DYN520 ALL COORDINATES	012401
C	IN STATE VECTOR Y INITIALLY TAKEN AS INDEPENDENT	012402
C	IV = IN LINEAR EQUIVALENCE(INDEP(I),IV(I))	012403
C		012404
C	*****	012405
C		012406
C		012407
C		012408
C		012409
C	RETURN	012410
C	END	012411

C	MSTEEETAABCEMTSSSSMDOBGA IENPRENNGOOLSELECTORDDCLDUTPIEORLYCTTT	012504
C	LLGNRR D U312340TUMRPNATECS IDDD OHAQNV IAUHSRETQTTYPUEI	012505
C	RVPKRTL G O 0000M BDEPR LP DCLT TORU R MMST E2 EE M	012506
C	NIPLUT *	012507
C	NUMS *	012508
C	NYPLCT *	012509
C	PAGEHD * * * * *	012510
C	PLOTSS * *	012511
C	PLCTWR *	012512
C	PLTCAR *	012513
C	PRNTOL *	012514
C	PR3 *	012515
C	QRCCN *	012516
C	QDRVR *	012517
C	QR2 *	012518
C	READ * * *	012519
C	READIV * *	012520
C	REVADC* *	012521
C	REVISE *	012522
C	RKACAM *	012523
C	RLOCUS *	012524
C	RLPLOT *	012525
C	RMVZRC *	012526
C	RCTCH *	012527
C	ROTCS *	012528
C	RUTTR ***	012529
C	RTAFE *	012530
C	RTCP *	012531
C	RWRITE *	012532
C	SATE *****	012533
C	SCALE *	012534
C	SFREQ2 *	012535
C	SFAFTT *	012536
C	SIFT *	012537
C	SKEWV3 *** ** *	012538
C	SPLCT *	012539
C	START *	012540
C	STORE *	012541
C	SUBCIA *	012542
C	TFLPY *	012543
C	AEBYBCCCCCCCCDDDEFFGGIILLLLLMMMMNNNPPQQRRRRRRRRRRSSSTTTWWY	012544
C	SADHSCORRRRRRRYYYYNI IERNNIPTTTAGRSSIUYALRRREEKLLDODTWFTFFCRRO	012545
C	IKOJOMNEEEEEENNNNGNNTVTVNRAGRIEIMMPMPGTNCUAAAUPTTTARRRLAPTRI IO	012546
C	MSTEEETAABCEMTSSSSMDOBGA IENPRENNGOOLSELECTORDDCLDUTPIEORLYCTTT	012547
C	LLGNRR D U312340TUMRPNATECS IDDD OHAQNV IAUHSRETQTTYPUEI	012548
C	RVPKRTL G O 0000M BDEPR LP DCLT TORU R MMST E2 EE M	012549
C	TFTYPE *	012550
C	TCRCUE *	012551
C	TRFE (NONE)	012552
C	TTF *	012553
C	UNITY *	012554
C	WRITE * ***** *	012555
C	WHITES**** * **** * *	012556
C	WRITIM (NONE)	012557
C	WRITIS* ** ** *** *	012558
C	WTAPE *	012559
C	YCOT *	012560
C	YCCTL *	012561
C	ZERG * ** ** * *	012562
C	DATE *	012563

```

C      SECOND                      * *                                012564
C      TIME                        *                                012565
C      BCXG                        * *                                012566
C      EXITG                       * *                                012567
C      GRIDG                       * *                                012568
C      LABELG                      * *                                012569
C      LEGNDG                      * *                                012570
C      LINESG                      * *                                012571
C      MCODEG                      * *                                012572
C      NUMERG                      * *                                012573
C      OBJCTG                      * *                                012574
C      PAGEG                       * *                                012575
C      FGRIDG                      * *                                012576
C      PLABELG                     * *                                012577
C      FLINEG                      * *                                012578
C      PSETG                       * *                                012579
C      PSUBJG                      * *                                012580
C      RSETG                       * *                                012581
C      SETSMG                      * *                                012582
C      SFTUPG                      * *                                012583
C      SUBJEG                      * *                                012584
C      TEXTG                      * *                                012585
C      TITLG                       * *                                012586
C      AEBLBCCCCCCCCDDDEFFGGIILLLLLMMMMNNNNPPPIJRRRRRRRRRRSSSTTTWWY 012587
C      SAJHSCURRRRRRRYYYYNITERNNIPITTAGRSSIUYALRRKEEKLLDUDOTWPTFFCRRO 012588
C      IKGGMNELELEEENNNNNNNNIVIVNRACRIEIMMPMPJINJAAAOPITTAARLAPTRIID 012589
C      MSTEELTAABCEMTSSSSMDDBGAILNPRENNGCOLSLELURDDUCLUDTPIEORLYQITTT 012590
C      LLNNNF 0 0312340UMKPNATEGS 1000 OHAUNV IAGUHSRETQTTYPCEI 012591
C      RVPRTL 0 0 0000M BDLPR LP DCLT TDJ R MMST E2 EE M 012592
C      012593
C      012594
C      012595
C      012596
C      ***** 012597
C      012598
C      SUBROUTINE ADDT          DEBUG # 105 012599
C      USER DEFINED SUBROUTINE 012600
C      CALLS: 012601
C      (NONE UNLESS USER REQUESTED) 012602
C      CALLED BY: 012603
C      YDDT 012604
C      PURPOSE: 012605
C      TO DEFINE THE ACCELERATION OF COORDINATE IC WHICH IS 012606
C      RHEOLOGICALLY CONSTRAINED, THAT IS TO PROVIDE THE 012607
C      ALPHA DOUBLE DOT TERM IN EQUATION 11-6, VOL I A 012608
C      USER DEFINED ROUTINE 012609
C      ***** 012610
C      012611
C      SUBROUTINE ADDJ          DEBUG # 27 012612
C      UTILITY, MATRIX OPERATION 012613
C      CALLS: 012614
C      (NONE) 012615
C      CALLED BY: 012616
C      CREC 012617
C      CREC 012618
C      PURPOSE: 012619
C      MULTIPLY THREE MATRICES EACH BY A SCALAR AND ADD. 012620
C      FORM THE SUM: 012621
C      AZ = SA* A + SB* B + SC* C 012622
C      012623

```

```

C          WHERE                                012624
C          A,B,C ARE INPUT (NR,NC) MATRICES    012625
C          SA,SB,SC ARE INPUT SCALARS          012626
C          B,C ARE NOT DESTROYED, INPUT A IS   012627
C          AZ OUTPUT MATRIX RETURNED IN A ARRAY 012628
C                                              012629
C*****                                         012630
C          SUBROUTINE ADT                      012631
C          DEBUG # 104                        012632
C          USER DEFINED SUBROUTINE            012633
C          CALLS:                             012634
C          (NONE UNLESS USER REQUESTED)       012635
C          CALLED BY:                         012636
C          FINDU                              012637
C          PURPOSE:                           012638
C          TO DEFINE THE VELOCITY OF COORDINATE IC WHICH IS 012639
C          RHEOMICALLY CONSTRAINED, THAT IS TO PROVIDE THE 012640
C          ALPHA DOT TERM IN EQUATION 11-5 VOL 1. A USER 012641
C          DEFINED ROUTINE                     012642
C          CAUTION TO AVOID EXTRANEIOUS IMPLUSE EFFECTS 012643
C          INTEGRATE ALPHA DOUBLE DOT TO GET ALPHA DOT 012644
C          PROBLEMS CAN ARISE DUE TO NUMERICAL INTEGRATION 012645
C                                              012646
C*****                                         012647
C          SUBROUTINE ALPHAA                   012648
C          DEBUG # 95                          012649
C          UTILITY, MATRIX OPERATION           012650
C          CALLS:                             012651
C          (NONE)                             012652
C          CALLED BY:                         012653
C          (NONE)                             012654
C          PURPOSE:                           012655
C          MULTIPLY SCALAR SA BY MATRIX A, RETURN Z 012656
C          Z = SA*A                            012657
C          A IS NOT DESTROYED                  012658
C                                              012659
C*****                                         012660
C          SUBROUTINE ASIMLR                   012661
C          DEBUG # 65                          012662
C          MAIN LINE OPERATION                 012663
C          CALLS:                             012664
C          ZERO WRITIS WRITES                012665
C          REVADD FINDT WRITE                 012666
C          MULTA GAUSSI MULTB                 012667
C          CALLED BY:                         012668
C          DYN340                             012669
C          PURPOSE:                           012670
C          ESTABLISH TRANSFORMED PARTIAL DERIVATIVE MATRIX 012671
C          BY PERFORMING A SIMILARITY TRANSFORMATION TO 012672
C          EXCHANGE PLANT STATE VARIABLES Y FOR SENSOR SIGNALS 012673
C          XSS AND CONTROL SYSTEM VARIABLES DELTA FOR TORQUE 012674
C          SIGNALS B. SEE SECTION 'EXCHANGE OF VARIABLES ' 012675
C          PAGE 59 VOL 1 FOR BACKGROUND THEORY 012676
C                                              012677
C*****                                         012678
C          SUBROUTINE ASUR                     012679
C          DEBUG # 118                        012680
C          UTILITY, MATRIX OPERATION           012681
C          CALLS:                             012682
C          (NONE)                             012683

```

C	CALLED BY:		012684
C	DYNB40		012685
C	PURPOSE:		012686
C	COMPUTE SQUARE OF THE MATRIX A, RETURNS A**2 IN B		012687
C	B = A*A		012688
C	A IS NOT DESTROYED		012689
C			012690
C	*****		012691
C			012692
C	SUBROUTINE DABT	DEEUG # 96	012693
C		UTILITY, MATRIX OPERATION	012694
C	CALLS:		012695
C	(NONE)		012696
C	CALLED BY :		012697
C	(NONE)		012698
C	PURPOSE:		012699
C	FORM THE SPECIAL MATRIX TRIPLE PRODUCT		012700
C	Z = B*A*B**T		012701
C	A MATRIX (NCB,NCB) SYMMETRIC		012702
C	B MATRIX (NRB,NCV)		012703
C	Z MATRIX (NRB,NRB) WILL BE SYMMETRIC		012704
C	NOTHING DESTROYED		012705
C			012706
C	*****		012707
C			012708
C	SUBROUTINE BAKSLV	DEEUG # 52	012709
C		UTILITY, MATRIX OPERATION	012710
C	CALLS:		012711
C	WRITES		012712
C	CALLED BY:		012713
C	YDOT		012714
C	PURPOSE:		012715
C	SPECIAL OPERATION TO USE DECOMPOSED BMB AND D		012716
C	MATRICES OBTAINED VIA DCUM2 ALONG WITH THE V MATRIX		012717
C	COMPUTED IN YDOT TO COMPUTE LAGRANGE MULTIPLIERS		012718
C			012719
C	*****		012720
C			012721
C	SUBROUTINE BDOTOP	DEEUG # 51	012722
C		MAIN LINE OPERATION	012723
C	CALLS:		012724
C	WRITES	SKEWVS	012725
C	MULTAD		012726
C	MULTS		012727
C	CALLED BY:		012728
C	YDOT		012729
C	PURPOSE:		012730
C	GET THE LOWER CASE B DOT MATRIX IN EQUATION II-6,		012731
C	REFER TO APPENDIX C VOL 1 FOR SUPPORTING THEORY		012732
C	ALSO PAGES 36-38, VOL 1		012733
C			012734
C	*****		012735
C			012736
C	SUBROUTINE BHGENR	DEEUG # 45	012737
C		MAIN LINE OPERATION	012738
C	CALLS:		012739
C	MULTS		012740
C	SKEWVS		012741
C	WRITES		012742
C	CALLED BY:		012743

C	DYNS40	012744
C	YDOT	012745
C	PURPOSE:	012746
C	USED TO COMPUTE THE ELEMENTS OF THE GH ARRAY THIS	012747
C	ARRAY DEFINES HINGE POINT KINEMATICS AND IS DEFINED	012748
C	IN VOL 1 BY EQUATIONS 11-82 AND 11-83	012749
C		012750
C		012751
C	*****	012752
C		012753
C	SUBROUTINE DSGENR	012754
C	DEBUG # 47	012755
C	MAIN LINE OPERATION	012756
C	CALLS:	012757
C	MULT3	012758
C	SKEW3	012759
C	CALLED BY:	012760
C	YDOT	012761
C	PURPOSE:	012762
C	COMPUTE THE ELEMENTS OF THE GS ARRAY THIS DEFINES	012763
C	THE KINEMATIC COEFFICIENTS FOR ALL SENSOR POINT	012764
C	REFERENCE FRAMES, SEE PAGE 17 VOL 11	012765
C		012766
C	*****	012767
C		012768
C	SUBROUTINE BTABA	012769
C	DEBUG # 35	012770
C	UTILITY, MATRIX OPERATION	012771
C	CALLS:	012772
C	(NONE)	012773
C	CALLED BY:	012774
C	MSMODC	012775
C	PURPOSE:	012776
C	FORM THE PARTICULAR MATRIX TRIPLE PRODUCT	012777
C	$AZ = B^{*T} * A * B$	012778
C	INPUT MATRICES A AND B, OUTPUT AZ. MATRIX A	012779
C	IS DESTROYED	012780
C		012781
C	*****	012782
C		012783
C	SUBROUTINE CANCEL	012784
C	DEBUG # 78	012785
C	UTILITY, NON-MATRIX OPER.	012786
C	CALLS:	012787
C	(NONE)	012788
C	CALLED BY:	012789
C	DYNS40	012790
C	PURPOSE:	012791
C	TO CANCEL OUT OF THE ROOTS ARRAY & THE ROOTS OF THE	012792
C	TRANSFER FUNCTION WHICH ARE COMMON TO BOTH	012793
C	NUMERATOR AND DENOMINATOR	012794
C		012795
C	*****	012796
C		012797
C	SUBROUTINE COMMENT	012798
C	DEBUG # 7	012799
C	UTILITY, DATA CONTROL, I/O	012800
C	CALLS:	012801
C	PAGEHD	012802
C	CALLED BY:	012803
C	MAIN	
C	PURPOSE:	
C	TO READ IN AND OUTPUT ON THE FIRST PAGE OF THE	
C	OUTPUT DATA ANY NUMBER OF USER WRITTEN COMMENT CARDS	

C		C12804
C	*****	012805
C		012806
C	SUBROUTINE CNTRL	012807
C	DEBUG # 107	012808
C	USER DEFINED SUBROUTINE	012809
C	CALLS:	012810
C	TFPLY ZERO	012811
C	READ (OTHERS WHICH ARE USER REQUESTED)	012812
C	WRITE	012813
C	CALLED BY:	012814
C	TORQUE	012815
C	PURPOSE:	012816
C	1)ACCEPT TRANSFER FUNCTION DESCRIPTION OF LINEAR	012817
C	CONTROL SYSTEM	012818
C	2)DEFINE ALL AUXILIARY DIFFERENTIAL EQUATIONS	012819
C	REQUIRED FOR SIMULATION PURPOSES (USER DEFINED)	012820
C	*****	012821
C		012822
C	SUBROUTINE CREA	012823
C	DEBUG # 21	012824
C	UTILITY, DATA PROCESSING	012825
C	CALLS:	012826
C	ZERO WRITE	012827
C	SATB	012828
C	FETCH	012829
C	CALLED BY:	012830
C	MSMODL	012831
C	PURPOSE:	012832
C	MAKES USE OF LUMPED PARAMETER DATA TO COMPUTE THE	012833
C	ALPHA COEFFICIENTS IN EQ 11-56 WHICH ARE ASSOCIATED	012834
C	WITH THE COMPUTATION OF THE CHANGE IN STATIC MASS	012835
C	MOMENT DUE TO FLEXIBLE BODY DEFORMATION, SEE ALSO	012836
C	APPENDIX-A VOL 1 EQS. A-12 TO A-20	012837
C		012838
C	*****	012839
C		012840
C	SUBROUTINE CREADD	012841
C	DEBUG # 19	012842
C	UTILITY, DATA PROCESSING	012843
C	CALLS:	012844
C	WRITE	012845
C	FETCH	012846
C	SATB	012847
C	CALLED BY:	012848
C	MSMODL	012849
C	PURPOSE:	012850
C	MAKES USE OF LUMPED PARAMETER DATA TO COMPUTE THE	012851
C	B AND A COEFFICIENTS WHICH ARE INDEPENDENT OF	012852
C	DEFORMATION AND DEFINE THE INERTIAL COUPLING BETWEEN	012853
C	6 ROTATION AND TRANSLATION COORDINATES AND THE	012854
C	IRGFLX(N) MODAL DISPLACEMENT COORDINATES,	012855
C	SEE EQ 11-67	012856
C		012857
C	*****	012858
C		012859
C	SUBROUTINE CREB	012860
C	DEBUG # 22	012861
C	UTILITY, DATA PROCESSING	012862
C	CALLS:	012863
C	ZERO SATB	

C		WRITE	012864
C		FETCH	012865
C	CALLED BY		012866
C		MSMODL	012867
C	PURPOSE:		012868
C		MAKES USE OF LUMPED PARAMETER DATA TO COMPUTE THE	012869
C		B COEFFICIENTS IN EQ II-88 WHICH ARE ASSOCIATED	012870
C		WITH THE COMPUTATION OF THE CHANGE IN THE BODY'S	012871
C		INERTIA TENSOR DUE TO TERMS LINEAR IN FLEXIBLE BODY	012872
C		DEFORMATION, SEE APPENDIX A VOL I EQS A-21 TO A-26	012873
C			012874
C			012875
C		*****	012876
C			012877
C	SUBROUTINE CREC	DEBUG # 23	012878
C		UTILITY, DATA PROCESSING	012879
C	CALLS:		012880
C		FETCH ZERO	012881
C		SATS WRITE	012882
C		ADDS	012883
C	CALLED BY:		012884
C		MSMODL	012885
C	PURPOSE:		012886
C		MAKES USE OF LUMPED PARAMETER DATA TO COMPUTE THE	012887
C		C COEFFICIENTS IN EQ II-88 WHICH ARE ASSOCIATED	012888
C		WITH THE COMPUTATION OF THE CHANGE IN THE COUPLING	012889
C		COEFFICIENTS BETWEEN ROTATION AND DISPLACEMENT	012890
C		COORDINATES DUE TO TERMS LINEAR IN FLEXIBLE BODY	012891
C		DEFORMATION, SEE APPENDIX A VOL I EQS A-27,28,29	012892
C		ALSO COMPUTES THE C COEFFICIENTS IN EQ II-89 WHICH	012893
C		ARE ASSOCIATED WITH THE COMPUTATION OF THE CHANGE	012894
C		IN THE BODY'S INERTIA TENSOR DUE TO TERMS QUADRATIC	012895
C		IN FLEXIBLE BODY DEFORMATION, SEE APPENDIX A VOL I	012896
C		EQS A-30 TO A-35	012897
C			012898
C		*****	012899
C			012900
C	SUBROUTINE CREC	DEBUG # 20	012901
C		UTILITY, DATA PROCESSING	012902
C	CALLS:		012903
C		WRITE ADDS	012904
C		FETCH	012905
C		SATS	012906
C	CALLED BY:		012907
C		MSMODL	012908
C	PURPOSE:		012909
C		MAKES USE OF LUMPED PARAMETER DATA TO COMPUTE THE	012910
C		E COEFFICIENTS WHICH ARE INDEPENDENT OF DEFORMATION	012911
C		THESE DEFINE THE MODAL MASS MATRIX WHICH IS DIAGONAL	012912
C		IF THE MODES ARE ORTHOGONAL SEE EQ II-67 MODES MAY	012913
C		BE NON-ORTHOGONAL	012914
C			012915
C			012916
C		*****	012917
C			012918
C	SUBROUTINE CREMOD	DEBUG # 18	012919
C		UTILITY, DATA PROCESSING	012920
C	CALLS:		012921
C		FETCH	012922
C		SATS	012923

C	WRITE	012924
C	CALLED BY:	012925
C	MSMDOL	012926
C	PURPOSE:	012927
C	MAKES USE OF LOADED PARAMETER DATA TO COMPUTE THE	012928
C	FLEXIBLE BODY'S UNDEFORMED INERTIA TENSOR, STATIC	012929
C	MASS MOMENT AND TOTAL MASS, SEE LU 11-67	012930
C		012931
C		012932
C	*****	012933
C		012934
C	SUBROUTINE CRET3 DEBUG # 17	012935
C	UTILITY, DATA PROCESSING	012936
C	CALLS:	012937
C	FETCH	012938
C	STORE	012939
C	CALLED BY:	012940
C	MSMDOL	012941
C	PURPOSE:	012942
C	TO FETCH RAW FLEXIBLE BODY DATA FROM NTAPE2 AND TO	012943
C	CREATE THE FUNCTIONS WHICH MUST BE INTEGRATED OVER	012944
C	THE BODY'S VOLUME TO DEFINE RESULTANT MODE DEPENDENT	012945
C	INERTIA PARAMETERS	012946
C		012947
C	*****	012948
C		012949
C	SUBROUTINE DCOM2 DEBUG # 42	012950
C	UTILITY, MATRIX OPERATION	012951
C	CALLS:	012952
C	(NONE)	012953
C	CALLED BY:	012954
C	YDOT	012955
C	PURPOSE:	012956
C	DECOMPOSES A TO FORM AN UPPER TRIANGULAR MATRIX	012957
C	WITH ONES ON DIAGONAL AND ZEROS BELOW SUCH THAT	012958
C	$A = U * T * D * U$	012959
C	A REAL, SQUARE, AND SYMMETRIC. ROUTINE ACTUALLY	012960
C	LEAVES GARBAGE ON DIAGONAL AND BELOW	012961
C	USED IN CONJUNCTION WITH BAKSLV TO OBTAIN	012962
C	LAGRANGE MULTIPLIERS	012963
C		012964
C	*****	012965
C		012966
C	SUBROUTINE DCCRT DEBUG # 70	012967
C	UTILITY, NON-MATRIX OPER.	012968
C	CALLS:	012969
C	(NONE)	012970
C	CALLED BY:	012971
C	DYN540	012972
C	PURPOSE:	012973
C	TAKES COMPLEX ROOTS (RR + I*RI) FOUND IN QR ROUTINES	012974
C	PLACES REAL AND ZERO ROOTS IN RLRT ARRAY,	012975
C	COUNTS THE NUMBER OF ZEROS (NZ), AND PLACES	012976
C	COMPLEX PAIRS IN CMPR ARRAY, (KC=NUMBER OF PAIRS)	012977
C		012978
C	*****	012979
C		012980
C	SUBROUTINE DEF1 DEBUG # 112	012981
C	DEFINITIONS	012982
C	CALLS:	012983

C	(NONE)	012984
C	CALLED BY :	012985
C	(NONE)	012986
C	PURPOSE:	012987
C	DEFINE COMMON BLOCKS AND PARAMETERS STORED THEREIN	012988
C		012989
C	*****	012990
C		012991
C	SUBROUTINE DEF2 DEBUG # 113	012992
C	DEFINITIONS	012993
C	CALLS:	012994
C	(NONE)	012995
C	CALLED BY :	012996
C	(NONE)	012997
C	PURPOSE:	012998
C	DEFINE ALL SUBROUTINES AS TO PURPOSE	012999
C		013000
C	*****	013001
C		013002
C	SUBROUTINE DEF3 DEBUG # 114	013003
C	DEFINITIONS	013004
C	CALLS:	013005
C	(NONE)	013006
C	CALLED BY :	013007
C	(NONE)	013008
C	PURPOSE:	013009
C	DEFINE REQUIRED INPUT DATA SET UP AND FORMAT	013010
C		013011
C	*****	013012
C		013013
C	SUBROUTINE DEF4 DEBUG # 115	013014
C	DEFINITIONS	013015
C	CALLS:	013016
C	(NONE)	013017
C	CALLED BY :	013018
C	(NONE)	013019
C	PURPOSE:	013020
C	DEFINE OUTPUT DATA	013021
C		013022
C	*****	013023
C		013024
C	SUBROUTINE DEF5 DEBUG # 116	013025
C	DEFINITIONS	013026
C	CALLS:	013027
C	(NONE)	013028
C	CALLED BY:	013029
C	(NONE)	013030
C	PURPOSE:	013031
C	DEFINE ALL PARAMETERS COMPUTED IN DISCOS	013032
C		013033
C	*****	013034
C		013035
C	SUBROUTINE DFORMB DEBUG # 80	013036
C	UTILITY, NON-MATRIX OPER.	013037
C	CALLS:	013038
C	(NONE)	013039
C	CALLED BY:	013040
C	DYN540	013041
C	PURPOSE:	013042
C	TAKES REAL ROOTS DEFINED IN ARRAY RLRT AND COMPUTES	013043

```

C          ASSOCIATED TIME CONSTANTS FOR = -1.0/KLRT          C13044
C          TAKES COMPLEX PAIRS DEFINED IN ARRAY CMPR. COMPUTES C13045
C          FJC(L) = OMEGA          C13046
C          FJC(L-1) = ZETA          C13047
C          WHERE COMPLEX PAIR ARE ROOTS OF:          C13048
C           $S^2 + 2*ZETA*OMEGA*S + OMEGA^2 = 0$           C13049
C          ALSO FORMS PRODUCT OF ALL ROOTS FOR GODE GAIN          C13050
C          COMPUTATION          C13051
C          C13052
C          C13053
C          C13054
C          FUNCTION DIMAG(S)          DEBUG # 76          C13055
C          UTILITY, NON-MATRIX OPER.          C13056
C          CALLS:          C13057
C          (NONE)          C13058
C          CALLED BY:          C13059
C          RLOCUS          C13060
C          SFREQ2          C13061
C          PURPOSE:          C13062
C          DIMAG(S) = IMAGINARY PART OF COMPLEX NUMBER S          C13063
C          PRECISION REAL*8          C13064
C          C13065
C          C13066
C          C13067
C          FUNCTION DREAL(S)          DEBUG # 77          C13068
C          UTILITY, NON-MATRIX OPER.          C13069
C          CALLS:          C13070
C          (NONE)          C13071
C          CALLED BY:          C13072
C          RLOCUS          C13073
C          SFREQ2          C13074
C          PURPOSE:          C13075
C          DREAL(S) = REAL PART OF COMPLEX NUMBER S          C13076
C          PRECISION REAL*8          C13077
C          C13078
C          C13079
C          C13080
C          SUBROUTINE DYN510          DEBUG # 2          C13081
C          MAIN LINE EXECUTIVE          C13082
C          CALLS:          C13083
C          MCMODC      MCMODL      PAGEHD          C13084
C          READIM      READ      WRITES          C13085
C          WRITES      MRIGID          C13086
C          CALLED BY:          C13087
C          MAIN          C13088
C          PURPOSE:          C13089
C          TO BUILD THE SIMULATION MODEL FROM THE INPUT DATA          C13090
C          SEE FLOW CHART PAGE 23 VOL 11, THE FOLLOWING          C13091
C          INPUT DATA IS READ:          C13092
C          1) BASIC MODEL SIZES AND TOPOLOGY          C13093
C          2) BODY TYPE, NUMBER OF MODES          C13094
C          3) SENSOR POINT LOCATION          C13095
C          4) CONSTRAINT CONDITIONS          C13096
C          5) CONTIGUOUS HINGE FRAME DATA, ORIENTATION + RATE          C13097
C          6) MOMENTUM WHEEL DATA          C13098
C          7) TIME HISTORY DATA          C13099
C          8) PRINT, PLOT CONTROL          C13100
C          9) CONTROL SYSTEM DATA          C13101
C          10) GRAVITY GRADIENT DATA          C13102
C          11) CALL MRIGID TO READ RIGID BODY CHARACTERISTICS          C13103

```

```

C          12) CALL MSMDDL TO READ FLEXIBLE BODY LUMPED          013104
C          PARAMETER DATA                                     013105
C          13) CALL MSMDDC TO READ FLEXIBLE BODY CONSISTANT     013106
C          MASS PARAMETER DATA                                013107
C          WITH THIS DATA SOME DATA SET ARRAYS ARE COMPUTED IN 013108
C          DYN510. AN ECHO OF ALL INPUT DATA AND COMPUTED DATA 013109
C          IS GIVEN. SEE SUBROUTINES DEF3 AND DEF4 LISTED IN     013110
C          VOL 2                                                 013111
C                                                                013112
C*****                                                         013113
C                                                                013114
C          SUBROUTINE DYN520          DEBUG # 37                013115
C                                                                013116
C          CALLS:          MAIN LINE EXECUTIVE                 013117
C                                                                013118
C          RKACAM          YDOT          ROTDH                 013118
C          FINDU          BHGENR          PRNTOU                013119
C          LINEAR          ENGMOM                                     013120
C          PLOTWR          WRITIS                                     013121
C          CALLED BY:                                     013122
C          MAIN                                             013123
C          PURPOSE:                                     013124
C          1) READ NTAPE1 ON WHICH ALL TIME INDEPENDENT RIGID AND 013125
C          FLEXIBLE BODY DATA HAS BEEN WRITTEN.             013126
C          2) LOAD INITIAL CONDITIONS INTO STATE VECTOR.       013127
C          3) COMPUTE INITIAL KINEMATIC DATA (BHGENR,ROTDH)    013128
C          4) FIND INDEPENDENT COORDINATES (FINDU)              013129
C          5) GET TIME DERIVATIVE OF STATE VECTOR (YDOT)        013130
C          6) COMPUTE ENERGY AND MOMENTUM FOR BODIES AND TOTAL 013131
C          SYSTEM (ENGMOM)                                       013132
C          7) INTEGRATE NON-LINEAR EQUATIONS OF MOTION (RKACAM) 013133
C          8) LINEARIZE COUPLED EQUATIONS OF MOTION (LINEAR)    013134
C          9) OUTPUT DATA ON PRINTER OR PLOTTER (PRNTOU,PLOTWR) 013135
C                                                                013136
C*****                                                         013137
C                                                                013138
C          SUBROUTINE DYN530          DEBUG # 62                013139
C                                                                013140
C          CALLS:          MAIN LINE EXECUTIVE                 013140
C                                                                013141
C          PLTCAR                                     013142
C          PAGEHD                                     013143
C          CALLED BY:                                     013144
C          MAIN                                             013145
C          PURPOSE:                                     013146
C          READ DATA REQUIRED TO SETUP PLOT TAPE FOR SC4020     013147
C                                                                013148
C*****                                                         013149
C                                                                013150
C          SUBROUTINE DYN540          DEBUG # 64                013151
C                                                                013152
C          CALLS:          MAIN LINE EXECUTIVE                 013153
C                                                                013154
C          ASIMLR          NUMB          READIM          RWRITE          TTFE          013154
C          CANCEL          NIPLOT          RLOCUS          SFREQ2          WRITE          013155
C          DCONRT          PAGEHD          RLPLT          SIFT          ZERO          013156
C          LTRSP          GRDNR          RMVZRO          SPLJT          DFURMB          013157
C          NIPLOT          READ          RTOP          T-TYPE          013158
C          CALLED BY:                                     013159
C          MAIN                                             013160
C          PURPOSE:                                     013161
C          ALL ANALYSIS ASSOCIATED WITH THE LINEARIZED          013162
C          EQUATIONS DIRECTED BY DYN540                       013163

```

C	1) READ LINEARIZED PARTIAL DERIVATIVE MATRIX FROM	C13164
C	NIAPC2	013165
C	2) FIND EIGENFREQUENCIES OF LINEARIZED SYSTEM (ORDRVR)	013166
C	3) GO THROUGH SIMILARITY TRANSFORMATION TO EXCHANGE	013167
C	SOME STATE VARIABLES FOR SENSOR AND TORQUE	013168
C	SIGNALS (ASIMR)	013169
C	4) FIND EIGENFREQUENCIES OF TRANSFORMED EQUATIONS,	013170
C	SHOULD BE SAME AS ABOVE (ORDRVR)	013171
C	5) READ DATA CARD TO DEFINE TYPE OF ANALYSIS	013172
C	6) CALL FOR LINEAR TIME HISTORY RESPONSE (LTRESP)	013173
C	7) READ CODES TO DEFINE TYPE OF TRANSFER FUNCTION,	013174
C	DETERM WHICH VARIABLES, INPUT DISPLAY	013175
C	8) OBTAIN CHARACTERISTIC MATRIX AND CORRESPONDING	013176
C	INPUT COEFFICIENTS FOR TRANSFER FUNCTION (IFTYPE)	013177
C	9) OBTAIN ROOTS OF DENOMINATOR (ORDRVR)	013178
C	10) OBTAIN ROOTS OF NUMERATOR (NUMB)	013179
C	11) SET OF ROOTS ARRAY (SIFT, DCONR, OFORMB, RVVZRC,	013180
C	AND IIFE)	013181
C	12) CANCEL OUT ROOTS COMMON TO NUMERATOR AND	013182
C	DENOMINATOR (CANCUR)	013183
C	13) PERFORM FREQUENCY RESPONSE, DISPLAY DATA VIA	013184
C	MODE, NICHOLS, AND/OR NYQUIST (SPLOT, NIPLDT, NYPLDT)	013185
C	14) CONVERT ROOTS TO POLYNOMIAL (RIOP)	013186
C	15) PERFORM ROOT LOCUS ANALYSIS (RESCUS, RLPLDT)	013187
C	16) COMPUTE EIGENVECTORS OF CHARACTERISTIC MATRIX	013188
C	FOR SELECTED EIGENVALUES	013189
C	*****	013190
C	*****	013191
C	SUBROUTINE EIGVEC DEBUG # 117	013192
C	UTILITY, MATRIX OPERATION	013193
C	CALLS:	013194
C	(NONE)	013195
C	CALLED BY:	013196
C	DYN540	013197
C	PURPOSE:	013198
C	TO FIND THE EIGENVECTORS OF A NON-SYMMETRIC MATRIX	013199
C	BY A MODIFIED WILKINSON'S INVERSE ITERATION METHOD	013200
C	SUBROUTINE ACCEPTS MATRIX A, A**2 AND EIGENVALUE	013201
C	LAMBDA FROM WHICH IT COMPUTES EIGENVECTOR X OF	013202
C	A**X = LAMBDA**X	013203
C	AND THE TRANSPOSED EIGENVECTOR Y OF	013204
C	A**T**Y = LAMBDA**Y	013205
C	EIGENVECTORS ASSOCIATED WITH REPEATED EIGENVALUES	013206
C	CANNOT BE FOUND	013207
C	*****	013208
C	*****	013209
C	*****	013210
C	SUBROUTINE ENGMCM DEBUG # 56	013211
C	MAIN LINE OPERATION	013212
C	CALLS:	013213
C	MULTD	013214
C	MULTAD	013215
C	CALLED BY:	013216
C	DYN520	013217
C	PURPOSE:	013218
C	TO COMPUTE LINEAR AND ANGULAR MOMENTUM ALONG WITH	013219
C	KINETIC AND POTENTIAL ENERGY FOR EACH BODY AND	013220
C	FOR THE TOTAL SYSTEM	013221
C		013222
C		013223


```

C***** 013224
C 013225
C SUBROUTINE LOAD          DEBUG # 110      013226
C                                USER DEFINED SUBROUTINE 013227
C CALLS:                    013228
C      (NONE UNLESS USER REQUESTED) 013229
C CALLED BY:                013230
C      TORQUE                013231
C PURPOSE:                  013232
C      TO DEFINE THE AUXILIARY EQUATIONS WHICH WILL DEFINE 013233
C      THE SENSOR SIGNAL AND TORQUE SIGNAL EQUATIONS TO BE 013234
C      USED FOR STABILITY ANALYSIS (USER DEFINED) 013235
C                                013236
C***** 013237
C 013238
C SUBROUTINE EXTOR          DEBUG # 108      013239
C                                USER DEFINED SUBROUTINE 013240
C CALLS:                    013241
C      (NONE UNLESS USER REQUESTED) 013242
C CALLED BY:                013243
C      TORQUE                013244
C PURPOSE:                  013245
C      TO DEFINE ALL EXTERNAL FORCE/TORQUE LOADS. 013246
C      THESE ACT AT SENSOR POINTS 013247
C                                013248
C***** 013249
C 013250
C SUBROUTINE FETCH          DEBUG # 25        013251
C                                UTILITY, DATA CONTROL, I/O 013252
C CALLS:                    013253
C      (NONE)                013254
C CALLED BY:                013255
C      MSMODL      CRELE 013256
C      CRETS       CREA 013257
C      CREMCO      CREB 013258
C      CREADD      CREC 013259
C PURPOSE:                  013260
C      TO FETCH A PARTICULAR MATRIX FROM TAPE. THE TAPE IS 013261
C      ASSUMED TO BE WRITTEN IN BINARY AND THE RECORD 013262
C      NUMBER ASSOCIATED WITH THE MATRIX KNOWN 013263
C                                013264
C***** 013265
C 013266
C SUBROUTINE FINDT          DEBUG # 66        013267
C                                MAIN LINE OPERATION 013268
C CALLS:                    013269
C      ZERO        WRITIS 013270
C      PAGEND      WRITES 013271
C      REWADD      013272
C CALLED BY:                013273
C      ASIMLR      013274
C PURPOSE:                  013275
C      COMPUTE THE SIMILARITY TRANSFORMATION MATRIX R OF 013276
C      EQUATION III-23, VOL I. SEE PAGE 63 VOL I ALSO 013277
C                                013278
C***** 013279
C 013280
C SUBROUTINE FINDU          DEBUG # 38        013281
C                                MAIN LINE OPERATION 013282
C CALLS:                    013283

```

C	ADT	013284
C	WRITIS	013285
C	WRITES	013286
C	CALLED BY:	013287
C	DYN520	013288
C	YDOT	013289
C	PURPOSE:	013290
C	IN DYN520 CALL FINDU TO COMPUTE THE INDEPENDENT	013291
C	COMPONENTS OF THE U VECTORS AT T=0 FOR INSERTION	013292
C	INTO THE STATE VECTOR.	013293
C	IN YDOT CALL FINDU TO COMPUTE THE DEPENDENT	013294
C	COORDINATES AT EACH TIME STEP FOR INSERTION INTO	013295
C	THE STATE VECTOR	013296
C	ONLY INDEPENDENT COORDINATES ARE INTEGRATED	013297
C	DEPENDENT COORDINATES MUST BE COMPUTED IN FINDU	013298
C	*****	013299
C	*****	013300
C	SUBROUTINE FIT	013301
C	DEBUG # 75	013302
C	UTILITY, NON-MATRIX OPER.	013303
C	CALLS:	013304
C	(NONE)	013305
C	CALLED BY:	013306
C	RLOCOS	013307
C	PURPOSE:	013308
C	FIND THE SLOPE OF THE LEAST SQUARES FIT LINE THROUGH	013309
C	THE LOG OF THE COEFFICIENTS OF A POLYNOMIAL. USED	013310
C	FOR SCALING TO RETAIN NUMERICAL PRECISION	013311
C	*****	013312
C	*****	013313
C	SUBROUTINE FORMB	013314
C	DEBUG # 79	013315
C	CALLS:	013316
C	(NONE)	013317
C	CALLED BY:	013318
C	(NONE)	013319
C	PURPOSE:	013320
C	NEARLY THE SAME AS DFORMB BUT WITHOUT GAIN	013321
C	COMPUTATION	013322
C	*****	013323
C	*****	013324
C	SUBROUTINE GAUSS1	013325
C	DEBUG # 31	013326
C	UTILITY, MATRIX OPERATION	013327
C	CALLS:	013328
C	WRITES	013329
C	CALLED BY:	013330
C	MSMDC LTRSP	013331
C	ASIMLR	013332
C	NUM5	013333
C	PURPOSE:	013334
C	MATRIX INVERSION SUBROUTINE. RETURNS MATRIX INVERSE	013335
C	ALONG WITH PIVOT ELEMENTS USED TO OBTAIN IT. THESE	013336
C	MAY BE USED TO OBTAIN THE DETERMINANT OF THE MATRIX	013337
C	*****	013338
C	*****	013339
C	SUBROUTINE GETUMB	013340
C	DEBUG # 48	013341
C	MAIN LINE OPERATION	013342
C	CALLS:	013343

C	WRITES	013344
C	MLTSK	013345
C	CALLER BY:	013346
C	YDOT	013347
C	PURPOSE:	013348
C	COMPUTE	013349
C	$GMB = B * M^{**}(-1) * B^{**}T$	013350
C	OF EQUATION 11-6 VOL 1. SAVE ALSO	013351
C	$BM = B * M^{**}(-1)$	013352
C	BOTH USED IN LAGRANGE MULTIPLIER COMPUTATION	013353
C		013354
C	*****	013355
C		013356
C	SUBROUTINE GMISC	013357
C	DEBUG # 111	013358
C	USER DEFINED SUBROUTINE	013359
C	CALLS:	013360
C	(NONE UNLESS USER REQUESTED)	013361
C	CALLER BY:	013362
C	TORQUE	013363
C	PURPOSE:	013364
C	A CATCH-ALL SUBROUTINE (USER DEFINED) USED TO ADD	013365
C	TO THE R.H.S. OF THE EQUATIONS OF MOTION ANY	013366
C	ADDITIONAL CONTRIBUTIONS SUCH AS THERMALLY INDUCED	013367
C	DEFORMATION EFFECTS	013368
C	*****	013369
C		013370
C	SUBROUTINE GRVGRD	013371
C	DEBUG # 50	013372
C	MAIN LINE OPERATION	013373
C	CALLS:	013374
C	WRITES	013375
C	MULT3	013376
C	MULTAD	013377
C	CALLER BY:	013378
C	YDOT	013379
C	PURPOSE:	013380
C	COMPUTE THE CONTRIBUTION TO THE FORCE VECTOR FROM	013381
C	GRAVITY GRADIENT EFFECTS	013382
C	*****	013383
C		013384
C	SUBROUTINE INTAPE	013385
C	DEBUG # 9	013386
C	UTILITY, DATA CONTROL, I/O	013387
C	CALLS:	013388
C	PAGEHD	013389
C	CALLER BY:	013390
C	READ	013391
C	PURPOSE:	013392
C	INITIALIZE TAPE FOR SUBROUTINE #TAPE	013393
C	*****	013394
C		013395
C	SUBROUTINE INVINP	013396
C	DEBUG # 41	013397
C	UTILITY, MATRIX OPERATION	013398
C	CALLS:	013399
C	WRITES	014000
C	CALLER BY:	014001
C	YDOT	014002
C	PRINTOU	014003
C	PURPOSE:	

C	COMPUTE INVERSE OF SYMMETRIC MASS MATRIX	013404
C		013405
C	*****	013406
C		013407
C	SUBROUTINE KHINGE	013408
C	DEBUG # 100	
C	USER DEFINED SUBROUTINE	013409
C	CALLS:	013410
C	(NONE UNLESS USER REQUESTED)	013411
C	CALLED BY:	013412
C	TORQUE	013413
C	PURPOSE:	013414
C	USER DEFINED ROUTINE USED TO DEFINE FORCES AND	013415
C	TORQUES ACTING AT HINGES DUE TO SPRINGS AND	013416
C	DAMPERS	013417
C		013418
C	*****	013419
C		013420
C	SUBROUTINE LINEAR	013421
C	DEBUG # 40	
C	MAIN LINE OPERATION	013422
C	CALLS:	013423
C	YDOT	013424
C	WRITES	013425
C	CALLED BY:	013426
C	DYNMOD	013427
C	PURPOSE:	013428
C	USED TO OBTAIN THE LINEARIZED EQUATIONS OF MOTION	013429
C	USING THE THEORY DISCUSSED IN VOL I SECTION III	013430
C	'SYNTHESIS AND ANALYSIS OF THE LINEARIZED SYSTEM'	013431
C	THE PARTIAL DERIVATIVE MATRIX A IN	013432
C	$\dot{Y}(0) = AY$	013433
C	IS DERIVED AND WRITTEN ON NIAPE2	013434
C		013435
C	*****	013436
C		013437
C	SUBROUTINE LPLTWK	013438
C	DEBUG # 100	
C	UTILITY, DATA CONTROL, I/O	013439
C	CALLS:	013440
C	(NONE)	013441
C	CALLED BY:	013442
C	LTRESP	013443
C	PURPOSE:	013444
C	TO WRITE TAPE FOR PLOTTING LINEARIZED TIME	013445
C	HISTORY RESPONSE DATA	013446
C		013447
C	*****	013448
C		013449
C	SUBROUTINE LPRINT	013450
C	DEBUG # 101	
C	UTILITY, DATA CONTROL, I/O	013451
C	CALLS:	013452
C	PAGEND	013453
C	WRITES	013454
C	CALLED BY:	013455
C	LTRESP	013456
C	PURPOSE:	013457
C	TO PRINT LINEARIZED TIME HISTORY RESPONSE	013458
C		013459
C	*****	013460
C		013461
C	SUBROUTINE LTAPE	013462
C	DEBUG # 14	
C	UTILITY, DATA CONTROL, I/O	013463

C	CALLS:		013464
C		PAGEND	013465
C	CALLED BY:		013466
C		READ	013467
C		RTAPE	013468
C	PURPOSE:		013469
C		LIST THE HEADINGS OF MATRICES WRITTEN ON	013470
C		TAPE NUMBER NIAPE	013471
C			013472
C	*****		013473
C			013474
C	SUBROUTINE LTORQL	DEBUG # 103	013475
C		USER DEFINED SUBROUTINE	013476
C	CALLS:		013477
C		ZERO	013478
C		(OTHERS IF USER REQUESTED)	013479
C	CALLED BY:		013480
C		LTRESP	013481
C	PURPOSE:		013482
C		TO COMPUTE ANY EXTERNAL FORCES AND TORQUES ACTING	013483
C		ON THE SYSTEM FOR LINEARIZED TIME HISTORY ANALYSIS	013484
C		RESULTS TO BE PUT IN VTORQ BY USER, IN YDOTL DISCOS	013485
C		WILL THEN SET UP THE MATRIX EQUATIONS	013486
C		$YDT = A*Y + VTORQ$	013487
C		TO BE INTEGRATED IN LTRESP (NOTE UNITS OF VTORQ)	013488
C			013489
C	*****		013490
C			013491
C	SUBROUTINE LTRESP	DEBUG # 99	013492
C		MAIN LINE OPERATION	013493
C	CALLS:		013494
C		ZERO YDOTL LTORQL	013495
C		MULTI LPLTWR	013496
C		GAUSS1 LPRNT	013497
C	CALLED BY:		013498
C		DYNS40	013499
C	PURPOSE:		013500
C		TO MAKE USE OF THE LINEARIZED EQUATIONS TO COMPUTE	013501
C		THE LINEARIZED TIME DOMAIN RESPONSE, SEE PAGES	013502
C		85-86 OF VOL 1. OUTPUT DATA SENT TO BOTH	013503
C		PRINT (LPRNT) AND PLOT (LPLTWR) ROUTINES	013504
C			013505
C	*****		013506
C			013507
C	MAIN (DISCOS)	DEBUG # 1	013508
C		MAIN LINE EXECUTIVE	013509
C	CALLS:		013510
C		DYNS10 START	013511
C		DYNS20 COMMENT	013512
C		DYNS30	013513
C		DYNS40	013514
C	CALLED BY:		013515
C		NONE, THIS IS THE MAIN PROGRAM	013516
C	PURPOSE:		013517
C		MAIN EXECUTIVE DRIVING PROGRAM. SIZES OF VARIOUS	013518
C		DIMENSIONED ARRAY DEFINED HERE	013519
C		THESE MAY BE CHANGED VIA THE REDIMENSION PROGRAM	013520
C			013521
C	*****		013522
C			013523

```

C      SUBROUTINE MGEN                      DEBUG # 49                      013524
C                                          MAIN LINE OPERATION          013525
C      CALLS:                               013526
C          WRITES                            013527
C          MULT3                             013528
C          MULTAD                            013529
C      CALLED BY:                          013530
C          YDOT                             013531
C      PURPOSE:                             013532
C          TO COMPUTE THE DEFORMATION DEPENDENT MASS MATRIX 013533
C                                          013534
C*****                                013535
C      SUBROUTINE MLTFR                      DEBUG # 56                      013537
C                                          UTILITY. MATRIX OPERATION 013538
C      CALLS:                               013539
C          (NONL)                            013540
C      CALLED BY:                          013541
C          GETBMB                            013542
C      PURPOSE:                             013543
C          USE TO PERFORM MATRIX OPERATION 013544
C              C = A*B                       013545
C          WHERE                             013546
C              C(G,LM) = A(G,LE)*B(LE,LM) 013547
C          FOR SELECTED ROWS OF C (NOTHING DESTROYED) 013548
C                                          013549
C*****                                013550
C      SUBROUTINE MRIGID                     DEBUG # 28                      013551
C                                          MAIN LINE OPERATION          013552
C      CALLS:                               013553
C          ZERO      WRITES      PAGEHD      013554
C          ROTTR     WRITES      013555
C          SKEWV3     READ        013556
C      CALLED BY:                          013557
C          DYN510     013558
C      PURPOSE:                             013559
C          READ AND PROCESS RIGID BODY DATA FOR ALL RIGID 013560
C          BODIES, SEE FLOW CHART PAGE 24 VOL II, DATA INPUT 013561
C          DEF3 AND DATA OUTPUT DEF4 013562
C                                          013563
C*****                                013564
C      SUBROUTINE MSMODC                     DEBUG # 15                      013565
C                                          MAIN LINE OPERATION          013566
C      CALLS:                               013567
C          READ      SKEWV3      MULTA      013568
C          UTABA     PR3         ROTTR      013569
C          UNITY     ZERO        READIM      013570
C          GAUSS1    PAGEHD      WRITES      013571
C          REVISE    WRITES      013572
C      CALLED BY:                          013573
C          DYN510     013574
C      PURPOSE:                             013575
C          READ AND PROCESS FLEXIBLE BODY DATA FOR EACH 013576
C          FLEXIBLE BODY DESCRIBED WITH CONSISTANT MASS DATA 013577
C          SEE FLOW CHART PAGES 27-28 VOL II 013578
C          DATA INPUT SUB. DEF3, DATA OUTPUT SUB. DEF4. 013579
C          CHECKY APPENDIX A, VOL I 013580
C                                          013581
C                                          013582
C                                          013583

```

C*****		013584
C		013585
C	SUBROUTINE MSMODL	013586
C	DEBUG # 16	
C	MAIN LINE OPERATION	013587
C	CALLS:	013588
C	CREE CREA FETCH	013589
C	READ CRTJ CREC	013590
C	ROTH WRITE ZERO	013591
C	CREADU CREMUO WRITES	013592
C	WRITIS CREB	013593
C	CALLED BY:	013594
C	DYNSIO	013595
C	PURPOSE:	013596
C	READ AND PROCESS FLEXIBLE BODY DATA FOR EACH	013597
C	FLEXIBLE BODY DESCRIBED WITH LUMPED MASS DATA	013598
C	SEE FLOW CHART PAGES 25-26 VOL 11, DATA INPUT	013599
C	SUB. DEFJ, DATA OUTPUT SUB. DEF+, THEORY APPENDIX A	013600
C	VOL 1	013601
C		013602
C*****		013603
C		013604
C	SUBROUTINE MULTA	013605
C	DEBUG # 34	
C	UTILITY, MATRIX OPERATION	013606
C	CALLS:	013607
C	(NONE)	013608
C	CALLED BY:	013609
C	MSMODL	013610
C	ASIMLR	013611
C	PURPOSE:	013612
C	UTILITY ROUTINE TO MULTIPLY RECTANGULAR MATRICES	013613
C	AZ * B = Z	013614
C	INPUT AZ AND B, OUTPUT Z IN DUMMY ARRAY AZ	013615
C		013616
C*****		013617
C		013618
C	SUBROUTINE MULTAD	013619
C	DEBUG # 55	
C	UTILITY, MATRIX OPERATION	013620
C	CALLS:	013621
C	(NONE)	013622
C	CALLED BY:	013623
C	MGEN TORQUE	013624
C	GRVGRD LENGMOM	013625
C	BDOOTUP	013626
C	PURPOSE:	013627
C	FORM MATRIX OPERATION	013628
C	ZZ = Z + A*B	013629
C	A,B,Z INPUT MATRICES	013630
C	ZZ OUTPUT IN Z ARRAY	013631
C	ORIGINAL Z DESTROYED A AND B ARE NOT	013632
C		013633
C*****		013634
C		013635
C	SUBROUTINE MULTB	013636
C	DEBUG # 97	
C	UTILITY, MATRIX OPERATION	013637
C	CALLS:	013638
C	(NONE)	013639
C	CALLED BY:	013640
C	ASIMLR	013641
C	LTRESP	013642
C	PURPOSE:	013643

```

C          UTILITY ROUTINE TO MULTIPLY RECTANGULAR MATRICES      013644
C          A * BZ = Z      013645
C          INPUT A AND BZ, OUTPUT Z IN THE DUMMY ARRAY BZ      013646
C          *****      013647
C          *****      013648
C          SUBROUTINE MULTS      013649
C          DEBUG # 04      013650
C          UTILITY, MATRIX OPERATION      013651
C          CALLS:      013652
C          (NONE)      013653
C          CALLED BY:      013654
C          ROTTR      BSGENR      TORQUE      013655
C          ROTDR      MGEN      ENGMEM      013656
C          BSGENR      GRYGRD      YDOT      013657
C          ROTDS      BDOTUP      013658
C          PURPOSE:      013659
C          FORM GENERAL MATRIX PRODUCT OF RECTANGULAR MATRICES      013660
C          A(NRA,NRC) * B(NPB,NCB) = Z(NRA,NCB)      013661
C          INPUT A AND B, OUTPUT Z NOTHING DESTROYED      013662
C          *****      013663
C          *****      013664
C          SUBROUTINE NIPLCT      013665
C          DEBUG # 90      013666
C          UTILITY, DATA PROCESSING      013667
C          UTILITY, DATA CONTROL, I/O      013668
C          CALLS:      013669
C          (SC4020 PLCT ROUTINES)      013670
C          CALLED BY:      013671
C          DYN540      013672
C          PURPOSE:      013673
C          GENERATE NICHOLS PLOTS AMPLITUDE VS PHASE ON      013674
C          RECTANGULAR GRID, DATA IN /PSJEFF/      013675
C          *****      013676
C          *****      013677
C          *****      013678
C          SUBROUTINE NUMS      013679
C          DEBUG # 08      013680
C          MAIN LINE OPERATION      013681
C          CALLS:      013682
C          GAUSSJ      WRITE      013683
C          QDRVR      013684
C          SIFT      013685
C          CALLED BY:      013686
C          DYN540      013687
C          PURPOSE:      013688
C          MAKES USE OF THE REDUCED CHARACTERISTIC MATRIX AND      013689
C          CORRESPONDING INPUT COEFFICIENTS TO SET UP THE      013690
C          NUMERATOR FOR THE PARTICULAR TRANSFER FUNCTION      013691
C          REQUESTED AND TO CALL THE ROUTINES TO GET ITS ROOTS      013692
C          *****      013693
C          *****      013694
C          SUBROUTINE NYPLCT      013695
C          DEBUG # 91      013696
C          UTILITY, DATA PROCESSING      013697
C          UTILITY, DATA CONTROL, I/O      013698
C          CALLS:      013699
C          PLOTS      013700
C          (SC4020 PLCT ROUTINES)      013701
C          CALLED BY:      013702
C          DYN540      013703
C          PURPOSE:

```


C	GENERATE NYQUIST PLOTS GAIN VS PHASE ON	013704
C	POLAR COORDINATE GRID, DATA IN /PSTUFF/	013705
C		013706
C	*****	013707
C		013708
C	SUBROUTINE PAGEHD	013709
C	DEBUG # 8	
C	UTILITY, DATA CONTROL, I/O	013710
C	CALLS:	013711
C	TIME	013712
C	SECOND	013713
C	CALLED BY:	013714
C	DYNS10 READ PRNTQU SFREQ2	013715
C	READIM WRITE DYNS30 RLOCUS	013716
C	WRITIM LTAPE DYNS40 RWRITE	013717
C	COMENT MSMODC FINDT LPRNT	013718
C	INTAPE MRIGID TFLY	013719
C	PURPOSE:	013720
C	UTILITY ROUTINE, BRINGS UP NEW PAGE ON LINE-PRINTER	013721
C	AND PUTS HEADING AT TOP OF IT	013722
C		013723
C	*****	013724
C		013725
C	SUBROUTINE PLOTSS	013726
C	DEBUG # 94	
C	UTILITY, DATA PROCESSING	013727
C	CALLS:	013728
C	(NONE)	013729
C	CALLED BY:	013730
C	RLPLOT	013731
C	NYPLOT	013732
C	PURPOSE:	013733
C	SELECTS PLOT UPPER AND LOWER LIMITS FOR A 10 SQUARE	013734
C	LINEAR PLOT GRID	013735
C		013736
C	*****	013737
C		013738
C	SUBROUTINE PLOTWR	013739
C	DEBUG # 61	
C	UTILITY, DATA CONTROL, I/O	013740
C	CALLS:	013741
C	(NONE)	013742
C	CALLED BY:	013743
C	DYNS20	013744
C	PURPOSE:	013745
C	PUT ALL TIME HISTORY DATA ON NTAPES, PLOTS MAY BE	013746
C	MADE UP ANY OF THE VARIABLES WRITTEN ON NTAPES	013747
C		013748
C	*****	013749
C		013750
C	SUBROUTINE PLICAR	013751
C	DEBUG # 63	
C	UTILITY, DATA CONTROL, I/O	013752
C	CALLS:	013753
C	RSETMG SETSMG SETUPG	013754
C	TITLEG GRIDG LINESG	013755
C	PAGEG TEXTG SUBJEG	013756
C	LABELG LEGNDG	013757
C	NUMBKG CBJCTG	013758
C	CALLED BY:	013759
C	DYNS30	013760
C	PURPOSE:	013761
C	MAKES USE OF SC4020 SUBROUTINES WHICH ARE RESIDENT	013762
C	IN THE GSFC 16M-360 SYSTEM TO GENERATE PLOTS FOR	013763

C	OUTPUT DATA GENERATED VIA NON-LINEAR TIME RESPONSE	013764
C	ANALYSIS, THE DATA IS TAKEN OFF NTAPE3 IN DYN530	013765
C	PUT IN DUMMY ARRAY AND PASSED TO PLTCAR FOR PLOT	013766
C	GENERATION	013767
C		013768
C	*****	013769
C	SUBROUTINE PRINTOU	013770
C	DEBUG # 60	013771
C	UTILITY, DATA CONTROL, I/O	013772
C	CALLS:	013773
C	PACEND	013774
C	WRITLS	013775
C	SECUND	013776
C	CALLED BY:	013777
C	DYN540	013778
C	PURPOSE:	013779
C	GENERAL PRINT ROUTINE FOR NON-LINEAR TIME HISTORY	013780
C	RESPONSE	013781
C		013782
C	*****	013783
C		013784
C	SUBROUTINE PR3	013785
C	DEBUG # 36	013786
C	UTILITY, MATRIX OPERATION	013787
C	CALLS:	013788
C	(NONE)	013789
C	CALLED BY:	013790
C	MMMODC	013791
C	PURPOSE:	013792
C	TO FORM THE MATRIX PRODUCT	013793
C	$ZC = Z + S * C ** T * A * B$	013794
C	WHERE	013795
C	Z IS NCC,NCB INPLT DESTROYED, OUTPUT RETURNED HERE	013796
C	C IS NRA,NCC NOT DESTROYED	013797
C	A IS NRA,NCA NOT DESTROYED	013798
C	B IS NCA,NCB NOT DESTROYED	013799
C	S IS SCALAR NOT DESTROYED	013800
C		013801
C	*****	013802
C		013803
C	SUBROUTINE QRCON	013804
C	DEBUG # 85	013805
C	UTILITY, MATRIX OPERATION	013806
C	CALLS:	013807
C	QR2	013808
C	CALLED BY:	013809
C	QRORVR	013810
C	PURPOSE:	013811
C	TO CALL FOR THE QR TRANSFORMATION AND TO MAKE USE	013812
C	OF THE DATA COMPUTE THEREIN TO DEFINE THE ROOTS OF	013813
C	THE MATRIX (ALL ROOTS ZERO,REAL,COMPLEX PAIRS)	013814
C		013815
C	*****	013816
C		013817
C	SUBROUTINE QRORVR	013818
C	DEBUG # 34	013819
C	UTILITY, MATRIX OPERATION	013820
C	CALLS:	013821
C	QRCON	013822
C	SJJDIA	013823
C	WRITLS	013824
C	CALLED BY:	013825
C	DYN540	013826

```

C          NUMS                                013824
C      PURPOSE:                                013825
C          TO DO A MINIMAL AMOUNT OF DATA ORGANIZATION AND 013826
C          TO CALL THE ROUTINES THAT WILL COMPUTED THE COMPLEX 013827
C          EIGENVALUES OF THE N,N MATRIX A. THIS ROUTINE IS 013828
C          AVAILABLE IN EXTENDED REAL*15 PRECISION HOWEVER 013829
C          RUN TIME ON IBM-360 IS ORDERS OF MAGNITUDE SLOWER 013830
C          THAN REAL*8 PRECISION 013831
C          013832
C*****013833
C      SUBROUTINE QR2                                DEEUG # 87 013834
C          UTILITY, MATRIX OPERATION 013835
C      CALLS:                                013836
C          (NONE) 013837
C      CALLED BY:                                013838
C          QRCUN 013839
C      PURPOSE:                                013840
C          TO GO THROUGH THE QR-TRANSFORMATION AND RETURN TO 013841
C          QRCUN THE DATA NECESSARY FOR SPECIFICATION OF THE 013842
C          COMPLEX ROOTS 013843
C          013844
C*****013845
C      SUBROUTINE READ                                DEEUG # 10 013846
C          UTILITY, DATA CONTROL, I/O 013847
C      CALLS:                                013848
C          INTAPE    WRITE 013849
C          RTAPE     LTAPE 013850
C          PAGEHD    WTAPE 013851
C      CALLED BY:                                013852
C          DYN510    MSMODL    DYN540 013853
C          MSMODC    MRIGID    CONTRL 013854
C      PURPOSE:                                013855
C          READ MATRIX OF REAL NUMBERS FROM CARDS OR TAPE, 013856
C          AND PRINT IT, SEE COMMENT CARDS IN SUBROUTINE 013857
C          013858
C*****013859
C      SUBROUTINE READIM                                DEEUG # 3 013860
C          UTILITY, DATA CONTROL, I/O 013861
C      CALLS:                                013862
C          PAGEHD 013863
C      CALLED BY:                                013864
C          DYN510    DYN540 013865
C          MSMODC 013866
C      PURPOSE:                                013867
C          READ INTEGER MATRIX IN FORMA FORMAT FROM CARDS 013868
C          PRINT ECHO OF INPUT IF * IN COL 80 OF HEADER CARD 013869
C          013870
C*****013871
C      SUBROUTINE REVAOD                                DEEUG # 98 013872
C          UTILITY, MATRIX OPERATION 013873
C      CALLS:                                013874
C          (NONE) 013875
C      CALLED BY:                                013876
C          FINDT 013877
C          ASIMLR 013878
C      PURPOSE:                                013879
C          013880
C          013881
C          013882
C          013883

```

C	USED TO REARRANGE ROWS AND COLUMNS OF MATRICES	013884
C	SEE COMMENT CARDS IN SUBROUTINE REVADD	013885
C		013886
C	*****	013887
C		013888
C	SUBROUTINE REVISE DEBUG # 50	013889
C		013890
C	CALLS:	013891
C	(MON2)	013892
C	CALLED BY:	013893
C	45400C	013894
C	PURPOSE:	013895
C	REARRANGE ROWS AND COLUMNS OF A SQUARE	013896
C	RECTANGULAR MATRIX. INPUT MATRIX NOT DESTROYED	013897
C		013898
C	*****	013899
C		013900
C	SUBROUTINE RERUNS DEBUG # 74	013901
C		013902
C	CALLS:	013903
C	DREAL FIT	013904
C	DIMAG	013905
C	PAGEHD	013906
C	CALLED BY:	013907
C	DYN540	013908
C	PURPOSE:	013909
C	TO COMPUTE THE ROOT LOCUS OF THE REQUESTED TRANSFER	013910
C	FUNCTION, OUTPUT DATA STORED IN DAVED AND SAVEP	013911
C	ARRAYS OF /PSTUFF/	013912
C		013913
C	*****	013914
C		013915
C	SUBROUTINE RKADAM DEBUG # 59	013916
C		013917
C	CALLS:	013918
C	YDOT	013919
C	CALLED BY:	013920
C	DYN520	013921
C	PURPOSE:	013922
C	BASIC INTEGRATION PACKAGE EITHER FOURTH ORDER RUNGE-	013923
C	KUTTA OR ADAMS PREDICTOR CORRECTOR METHODS AVAILABLE	013924
C		013925
C	*****	013926
C		013927
C	SUBROUTINE RLPLDT DEBUG # 92	013928
C		013929
C	UTILITY, DATA PROCESSING	013930
C	UTILITY, DATA CONTROL, I/O	013931
C	CALLS:	013932
C	PLOTSS	013933
C	(SC4020 PLOT ROUTINES)	013934
C	CALLED BY:	013935
C	DYN540	013936
C	PURPOSE:	013937
C	GENERATE ROOT LOCUS PLOT FOR A SINGLE ROOT	013938
C	DATA IN /PSTUFF/	013939
C		013940
C	*****	013941
C		013942
C	SUBROUTINE RMVZRU DEBUG # 71	013943
C		013944
C	UTILITY, NON-MATRIX OPER.	013945

C	CALLS:		013944
C		(NONE)	013945
C	CALLED BY:		013946
C		DYNS40	013947
C	PURPOSE:		013948
C		REMOVE REAL ZEROS FROM REAL ROOT ARRAY PRIOR	013949
C		TO CALL TO TUFF	013950
C			013951
C	*****		013952
C			013953
C	SUBROUTINE ROTDH	DEBUG # 44	013954
C		MAIN LINE OPERATION	013955
C	CALLS:		013956
C		MULT3	013957
C		ROTTR	013958
C		WRIT3	013959
C	CALLED BY:		013960
C		DYNS20	013961
C		YDOT	013962
C	PURPOSE:		013963
C		TO COMPUTE HINGE FRAME TRANSFORMATIONS AND VELOCITY	013964
C		TRANSFORMATION MATRICES. THE ELEMENTS OF THE ARRAYS	013965
C		DH,RH,PIN,RUL AND DOL ARE COMPUTED HERE	013966
C			013967
C	*****		013968
C			013969
C	SUBROUTINE ROTDS	DEBUG # 46	013970
C		MAIN LINE OPERATION	013971
C	CALLS:		013972
C		MULT3	013973
C		ROTTR	013974
C	CALLED BY:		013975
C		YDOT	013976
C	PURPOSE:		013977
C		TO COMPUTE SENSOR POINT FRAME TRANSFORMATION MATRIX	013978
C		THE ELEMENTS OF THE ARRAYS RS AND DS ARE COMPUTED	013979
C		HERE	013980
C			013981
C	*****		013982
C			013983
C	SUBROUTINE ROTTR	DEBUG # 32	013984
C		MAIN LINE OPERATION	013985
C	CALLS:		013986
C		MULT3	013987
C	CALLED BY:		013988
C		MSMODC ROTDH	013989
C		MSMODL ROTDS	013990
C		MKRGID	013991
C	PURPOSE:		013992
C		TO COMPUTE ANY ONE OF THE TWELVE TYPES OF EULER	013993
C		ANGLE TRANSFORMATION MATRICES AND/OR THE PI INVERSE	013994
C		VELOCITY TRANSFORMATIONS. FOR THEORY SEE APPENDIX B,	013995
C		VOL 1. ALSO SEE COMMENT CARDS ROTTR	013996
C			013997
C	*****		013998
C			013999
C	SUBROUTINE RTAPE	DEBUG # 11	014000
C		UTILITY, DATA CONTROL, I/O	014001
C	CALLS:		014002
C		LTAPE	014003

C	CALLED BY:		014004
C	READ		014005
C	PURPOSE:		014006
C	READ MATRIX A. SIZE(NRA,NCA) FROM TAPE ALONG WITH		014007
C	ROW AND COLUMN LENGTHS. SEE COMMENT CARDS IN RTAPE		014008
C	*****		014009
C			014010
C	SUBROUTINE RTOP	DEEUG # 81	014011
C		UTILITY. NJN-MATRIX OPER.	014012
C	CALLS:		014013
C	(NONE)		014014
C	CALLED BY:		014015
C	DYNS40		014016
C	PURPOSE:		014017
C	TO TAKE THE NUMERATOR AND DENOMINATOR ROOTS STORED		014018
C	IN THE ROOTS ARRAY R AND GENERATE THE CORRESPONDING		014019
C	POLYNOMIAL COEFFICIENTS. THE COEFFICIENTS ARE		014020
C	STORED IN ARRAY RX OF /LRD01/		014021
C	*****		014022
C			014023
C			014024
C	SUBROUTINE RWRITE	DEEUG # 88	014025
C		UTILITY. DATA CONTROL, I/O	014026
C	CALLS:		014027
C	PAGEHD		014028
C	CALLED BY:		014029
C	DYNS40		014030
C	PURPOSE:		014031
C	PRINT OUT REAL AND IMAGINARY PARTS OF THE ROOTS		014032
C	OBTAINED AT DIFFERENT STEPS THROUGH THE STABILITY		014033
C	ANALYSIS		014034
C	*****		014035
C			014036
C			014037
C	SUBROUTINE SATB	DEEUG # 26	014038
C		UTILITY. MATRIX OPERATION	014039
C	CALLS:		014040
C	(NONE)		014041
C	CALLED BY:		014042
C	CRHMUD CRLE		014043
C	CRHADD CRCL		014044
C	CRHA CRCE		014045
C	PURPOSE:		014046
C	PERFORM THE MATRIX OPERATION		014047
C	ZZ = Z + S*A**T*B		014048
C	Z,A,B INPUT MATRICES		014049
C	S INPUT SCALAR		014050
C	ZZ OUTPUT MATRIX RETURNED IN Z ARRAY		014051
C	Z INPUT DESTROYED		014052
C	A,B,S NOT DESTROYED		014053
C	*****		014054
C			014055
C			014056
C	SUBROUTINE SCALE	DEEUG # 93	014057
C		UTILITY. DATA PROCESSING	014058
C	CALLS:		014059
C	(NONE)		014060
C	CALLED BY:		014061
C	SPLIT		014062
C			014063

C	PURPOSE:	014064
C	COMPUTE SCALE FACTOR AND SCALE LIMITS FOR PLOTTING	014065
C		014066
C	*****	014067
C		014068
C	SUBROUTINE SFEQ2	014069
C	DEBUG # 73	014070
C	UTILITY, NON-MATRIX OPER.	014071
C	CALLS:	014072
C	PAGEHD DREAL	014073
C	DIMAG	014074
C	CALLED BY:	014075
C	DYNS40	014076
C	PURPOSE:	014077
C	TO DETERMINE S-PLANE FREQUENCY RESPONSE. ALL DATA	014078
C	COMPUTED STORED IN /PSTUFF/ FOR PLOTTING	014079
C	MAKES USE OF TIME CONSTANTS, DAMPING ZETAS AND	014080
C	FREQUENCIES STORED IN ARRAYS FBRN,FBNF,FBR=FBRD	014081
C	AND FBDC ALONG WITH BUDE GAIN=GAIN	014082
C	*****	014083
C		014084
C	SUBROUTINE SHAFFT	014085
C	DEBUG # 109	014086
C	USER DEFINED SUBROUTINE	014087
C	CALLS:	014088
C	(NONE UNLESS USER REQUESTED)	014089
C	CALLED BY:	014090
C	TORQUE	014091
C	PURPOSE:	014092
C	TO DEFINE ALL TORQUES ACTING ON EMBEDDED	014093
C	SYMMETRIC WHEELS (USER DEFINED)	014094
C	*****	014095
C		014096
C	SUBROUTINE SIFT	014097
C	DEBUG # 72	014098
C	UTILITY, NON-MATRIX OPER.	014099
C	CALLS:	014100
C	(NONE)	014101
C	CALLED BY:	014102
C	DYNS40	014103
C	NUM5	014104
C	PURPOSE:	014105
C	SEARCH INPUT ARRAY A FOR NUMBERS LESS THAN TOLERANCE	014106
C	VALUE TOL, SET THESE EQUAL TO 0.0	014107
C	*****	014108
C		014109
C	SUBROUTINE SKEWV3	014110
C	DEBUG # 53	014111
C	UTILITY, MATRIX OPERATION	014112
C	CALLS:	014113
C	(NONE)	014114
C	CALLED BY:	014115
C	MSMODC BSGENR	014116
C	MRIGID BDCUP	014117
C	BHGENR TORQUE	014118
C	PURPOSE:	014119
C	UTILITY ROUTINE SKEW OPERATION	014120
C	0 V3 -V2	014121
C	SKV = -V3 0 V1	014122
C	V2 -V1 0	014123
C	INPUT (V1,V2,V3) OUTPUT SKV ARRAY	

```

C
C*****
C
C      SUBROUTINE SPSLOT          DEBUG # 39
C                                  UTILITY, DATA PROCESSING
C                                  UTILITY, DATA CONTROL, I/O
C      CALLS:
C          SCALE
C          (SCALING PLT ROUTINES)
C      CALLED BY:
C          DYIN40
C      PURPOSE:
C          GENERATE BODE PLOTS GAIN AND PHASE VS FREQUENCY
C          ON SEMI-LOG SCALE, DATA IN ZPTDF-7
C*****
C
C      SUBROUTINE START          DEBUG # 3
C                                  UTILITY, DATA CONTROL, I/O
C      CALLS:
C          EXITG
C          MDELEG
C          DATE
C      CALLED BY:
C          MAIN
C      PURPOSE:
C          INITIALIZE PLOTTING CAPABILITY, READS DATA TO BE PUT
C          AS HEADING ON EACH OUTPUT PAGE. IF TRONNG.FG.6HDEBUG
C          TWO EXTRA CARDS READ WHICH DEFINE THE SUBROUTINES
C          FOR WHICH THE DEBUG PRINT OPTION IS TO BE
C          EXERCISED
C*****
C
C      SUBROUTINE STORE          DEBUG # 24
C                                  UTILITY, MATRIX OPERATION
C                                  UTILITY, DATA CONTROL, I/O
C      CALLS:
C          (NONE)
C      CALLED BY:
C          CRETS
C      PURPOSE:
C          FORM MATRIX PRODUCT
C               $Z = A * B$ 
C          WHERE:
C              A IS (NA,NA) DIAGONAL STORED AS A VECTOR
C              B IS (NA,NCB) GENERAL
C          WRITE Z IN BINARY ON TAPE NTAPE
C*****
C
C      SUBROUTINE SUBDIA          DEBUG # 36
C                                  UTILITY, MATRIX OPERATION
C      CALLS:
C          (NONE)
C      CALLED BY:
C          QROKVR
C      PURPOSE:
C          TO PUT THE MATRIX A IN UPPER HESSENBURG FORM.
C          PERLIMINARY STEP IN OBTAINING COMPLEX ROOTS

```



```

C 014184
C ***** 014185
C 014186
C SUBROUTINE TPLY          DEBUG # 69          014187
C                                MAIN LINE OPERATION 014188
C CALLS: 014189
C        PAGEHD 014190
C CALLED BY: 014191
C        CONTRL 014192
C PURPOSE: 014193
C        TO CONVERT POLYNOMIAL TRANSFER FUNCTION EXPRESSIONS 014194
C        OF CONTROL LOOPS TO FIRST ORDER DIFFERENTIAL 014195
C        EQUATIONS, SEE VOL 1 PAGE 73 FOR THEORY. 014196
C        OUTPUT OF TPLY LOADED DIRECTLY INTO STATE VECTOR 014197
C        DERIVATIVE ARRAY YDT 014198
C 014199
C ***** 014200
C 014201
C SUBROUTINE TTYPE          DEBUG # 67          014202
C                                MAIN LINE OPERATION 014203
C CALLS: 014204
C        ZERO 014205
C CALLED BY: 014206
C        DYN540 014207
C PURPOSE: 014208
C        MAKES USE OF THE INPUTTED INTEGER CODES AND THE 014209
C        LINEARIZED SYSTEM EQUATIONS OF MOTION TO SET UP 014210
C        THE REDUCED CHARACTERISTIC MATRIX AND THE 014211
C        CORRESPONDING INPUT COEFFICIENTS FOR THE PARTICULAR 014212
C        TRANSFER FUNCTION REQUESTED, FOR THEORY SEE VOL 1, 014213
C        PAGES 73-78 014214
C 014215
C ***** 014216
C 014217
C SUBROUTINE TORQUE          DEBUG # 57          014218
C                                MAIN LINE OPERATION 014219
C CALLS: 014220
C        MULTS    CONTRL    WRITES    SKEWV3 014221
C        MULTAD   EXTOR     GMISC     014222
C        EQUAD    SHAFFT    KHINGE     014223
C CALLED BY: 014224
C        YDOT 014225
C PURPOSE: 014226
C        COMPUTE ALL GYROSCOPIC FORCES AND TORQUES ACTING 014227
C        ON THE SYSTEM, CALL USER ROUTINES WHICH DEFINE 014228
C        NON-GYROSCOPIC EFFECTS (SPRING, DASHPUTS, MOTORS) 014229
C        WHEEL CONTROL TORQUES, GRAVITY GRADIENT, ALSO 014230
C        CALL TO EQUAD DEFINES THE SENSOR AND TORQUE SIGNAL 014231
C        EQUATIONS FOR STABILITY ANALYSIS 014232
C 014233
C ***** 014234
C 014235
C SUBROUTINE TRFB          DEBUG # 82          014236
C CALLS: 014237
C        (NONE) 014238
C CALLED BY: 014239
C        (NONE) 014240
C PURPOSE: 014241
C 014242
C 014243

```

```

C*****
C
C      SUBROUTINE T1FF              DEBUG # 83
C                                  UTILITY, NON-MATRIX OPER.
C      CALLS:
C          (NONE)
C      CALLED BY:
C          DYN540
C      PURPOSE:
C          TAKES ALL REAL,ZEFC AND COMPLEX ROOTS OF NUMERATOR
C          AND DENOMINATOR ALONG WITH JOSE GAIN FOR PARTICULAR
C          TRANSFER FUNCTION AND PLACES THEM IN THE ROOTS
C          ARRAY R
C*****
C
C      SUBROUTINE UN1TY              DEBUG # 33
C                                  UTILITY, MATRIX OPERATION
C      CALLS:
C          (NONE)
C      CALLED BY:
C          MSMODC
C      PURPOSE:
C          CREATE UNIT DIAGONAL MATRIX
C*****
C
C      SUBROUTINE WR1TE              DEBUG # 12
C                                  UTILITY, DATA CONTROL, I/O
C      CALLS:
C          PAGE10
C      CALLED BY:
C          READ      CREE      ASIMLR
C          MSMODL    CREA      NUMS
C          CREMOD     CREB      DYN540
C          CREADD     CREC      CONTRL
C      PURPOSE:
C          UTILITY ROUTINE USED TO PRINT TWO DIMENSIONAL MATRIX
C          OF REAL NUMBERS IN FORMA FORMAT WITH LABELED HEADING
C          AT TOP OF NEW PAGE
C*****
C
C      SUBROUTINE WR1TES              DEBUG # 13
C                                  UTILITY, DATA CONTROL, I/O
C      CALLS:
C          (NONE)
C      CALLED BY:
C          DYN510    LINEAR    BDC10P    LPRINT
C          MSMODC    IN1NP     EAKSLV    FINDT
C          MSMODL    B10ENR    TORQUE     ASIMLR
C          MR1010    GETGMD    PRN100
C          FINDG     MS1EN     GAUSS1
C          YOUT      GRVORD     GRVORV
C      PURPOSE:
C          UTILITY ROUTINE TO PRINT TWO DIMENSIONAL ARRAY OF
C          REAL NUMBERS IN FORMA FORMAT.
C*****
C*****

```

```

014244
014245
014246
014247
014248
014249
014250
014251
014252
014253
014254
014255
014256
014257
014258
014259
014260
014261
014262
014263
014264
014265
014266
014267
014268
014269
014270
014271
014272
014273
014274
014275
014276
014277
014278
014279
014280
014281
014282
014283
014284
014285
014286
014287
014288
014289
014290
014291
014292
014293
014294
014295
014296
014297
014298
014299
014300
014301
014302
014303

```

```

C      SUBROUTINE WRITIM          DEBUG # 4          014304
C                                  UTILITY, DATA CONTROL, I/O 014305
C      CALLS:                      014306
C      PAGEHD                      014307
C      CALLED BY:                  014308
C      (NONE)                     014309
C      PURPOSE:                   014310
C      PULL UP NEW PAGE PRINT INTEGER ARRAY IN FORMA 014311
C      FORMAT WITH LABEL          014312
C                                  014313
C*****                          014314
C                                  014315
C      SUBROUTINE WRITIS          DEBUG # 5          014316
C                                  UTILITY, DATA CONTROL, I/O 014317
C      CALLS:                      014318
C      (NONE)                     014319
C      CALLED BY:                  014320
C      DYN510      DYN520      ASIMLR 014321
C      MSMODC      FINDU      014322
C      MSMODL      ROTDH      014323
C      MRIGID      FINDT      014324
C      PURPOSE:                   014325
C      PRINT TWO DIMENSIONAL INTEGER ARRAYS IN FORMA FORMAT 014326
C                                  014327
C*****                          014328
C                                  014329
C      SUBROUTINE WTAPE          DEBUG # 13         014330
C                                  UTILITY, DATA CONTROL, I/O 014331
C      CALLS:                      014332
C      (NONE)                     014333
C      CALLED BY:                  014334
C      READ      014335
C      PURPOSE:                   014336
C      WRITE MATRIX A IN BINARY ON TAPE, SEE COMMENT 014337
C      CARDS IN WTAPE            014338
C                                  014339
C*****                          014340
C                                  014341
C      SUBROUTINE YDOT          DEBUG # 39         014342
C                                  MAIN LINE OPERATION 014343
C      CALLS:                      014344
C      ADDT      BDOTOP      BFGENR      GAKSLV 014345
C      DCOM2     GETOMB      GRVGRD      BGENR 014346
C      FINDU     WRITES      MGEN      INVINP 014347
C      ROTDS     MULT3      ROTDH      TURQUE 014348
C      CALLED BY:                  014349
C      DYN520      014350
C      RKADAM      014351
C      LINEAR      014352
C      PURPOSE:                   014353
C      TO COMPUTE THE FIRST TIME DERIVATIVE OF THE 014354
C      STATE VECTOR Y, SEE COMMENT CARDS IN YDOT 014355
C                                  014356
C*****                          014357
C                                  014358
C      SUBROUTINE YDOTL          DEBUG # 102        014359
C                                  MAIN LINE OPERATION 014360
C      CALLS:                      014361
C      (NONE)                     014362
C      CALLED BY:                  014363

```

C	LTRESP	014364
C	PURPOSE:	014365
C	TO SET UP THE LINEARIZED TIME DOMAIN EQUATIONS OF	014366
C	MOTION FOR THE TOTAL SYSTEM. THESE ARE OF THE FORM	014367
C	$YDT = A*Y + B$	014368
C	THEY ARE INTEGRATED IN LTRESP.	014369
C	*****	014370
C		014371
C	SUBROUTINE ZERO	014372
C	DEBUG # 29	014373
C	UTILITY, MATRIX OPERATION	014374
C	CALLS:	014375
C	(NONE)	014376
C	CALLED BY:	014377
C	MSMODL CREC FIRST CONTRL	014378
C	MSMODC MRIGID TETYPE	014379
C	CREA CYS40 LTRESP	014380
C	CREB ASIMLR LTORQL	014381
C	PURPOSE:	014382
C	UTILITY ROUTINE TO ENTER 0.0 IN THE FIRST NR ROWS	014383
C	AND NC COLUMNS OF THE DUMMY ARRAY Z(KR,1)	014384
C	*****	014385
C		014386
C	SUBROUTINES DATE, TIME, AND SECOND	014387
C	PURPOSE:	014388
C	SYSTEM ROUTINES FOR DEFINING RUN TIME	014389
C	DATE AND TIME USED TO FIND ACTUAL TIME AND DATE	014390
C	OF RUN	014391
C	SECOND USED TO FIND ELAPSED CPU TIME AT VARIOUS	014392
C	CHECKPOINTS IN PROGRAM	014393
C	*****	014394
C		014395
C		014396
C	SUBROUTINES BOXG, EXITS, GRIDG, LABELG....ETC	014397
C	PURPOSE:	014398
C	SC4020 SYSTEM PLOTTING ROUTINES. THESE ARE RESIDENT	014399
C	IN THE GSFC IBM-360 SYSTEM	014400
C	*****	014401
C		014402
C		014403
C		014404
C		014405
C		014406
C		014407
C		014408
C		014409
C		014410
C	RETURN	014411
C	END	014412
C	SUBROUTINE DEF3	014413
C	DEBUG # 114	014414
C		014415
C		014416
C		014417

C		014418
C	***** DISCOS PROGRAM DATA STREAM *****	014419
C		014420
C	*****	014421
C		014422
C		014423
C	*****	014424
C*		014425
C*	MAIN PROGRAM	014426
C*		014427
C	*****	014428
C		014429
C	NIT = INPUT TAPE NUMBER	014430
C		014431
C	----- DENOTES WHERE DATA	014432
C	----- IS READ INTO PROGRAM	014433
C		014434
C		014435
C	-----999 CALL START	014436
C		014437
C	-----CALL CLEMENT	014438
C		014439
C	-----CALL DYN510	014440
C		014441
C	-----CALL DYN520	014442
C		014443
C		014444
C	IFLNER = LINEARIZATION FLAG = IPDATA(3)	014445
C	NOPLUT = PLUT CONTROL FLAG = IPDATA(2)	014446
C		014447
C	-----IF (IFLNER .EQ. 1) CALL DYN540	014448
C		014449
C	-----IF (NOPLUT .GT. 0) CALL DYN530	014450
C		014451
C	GO TO 999	014452
C		014453
C	END	014454
C		014455
C		014456
C		014457
C		014458
C		014459
C	*****	014460
C*		014461
C*	SUBROUTINE START - INPUT IDENTIFICATIONS	014462
C*		014463
C	*****	014464
C		014465
C		014466
C	-----READ(NIT,FORMAT = A6,4X,3A6) IRUNNO, (UNAME(I),I=1,3)	014467
C		014468
C	IRUNNO = RUN IDENTIFICATION NUMBER (5 CHARACTERS)	014469
C	UNAME = USERS NAME (18 CHARACTERS)	014470
C		014471
C	IRUNNO EQ 4HSTOP - TERMINATE THE RUN	014472
C	IRUNNO NE 4HSTOP - CONTINUE THE RUN	014473
C		014474
C	DEBUG(I) = .TRUE. IF DEBUG PRINT STATEMENTS CODED IN	014475
C	SUBROUTINE DEBUG # I ARE TO BE USED	014476
C	= .FALSE. IF NO DEBUG PRINTING DESIRED IN	014477

B-255

C	NODELTA = NUMBER OF ADDITIONAL DIFFERENTIAL EQS	014538
C	REQUIRED - CONTAINS CONTROL SYSTEM	014539
C	VARIABLES	014540
C		014541
C		014542
C	-----CALL READIM (ITOPOL, 2, NH, 2, NHMAX)	014543
C		014544
C	MATRIX SIZE 2 BY NH TO DESCRIBE TOPOLOGY	014545
C		014546
C	FOR THE JTH HINGE -	014547
C		014548
C	ITOPOL(1,J) = BODY N	014549
C	ITOPOL(2,J) = BODY M	014550
C	BODY N CONNECTED TO BODY M AT HINGE J	014551
C	ITOPOL(1,1) = 1 BY DEFINITION (BODY 1)	014552
C	ITOPOL(2,1) = 0 BY DEFINITION (INERTIAL REF)	014553
C	THE Q-FRAME IS ON BODY N	014554
C	THE P-FRAME IS ON BODY M	014555
C	MOTION OF THE Q-FRAME IS MEASURED RELATIVE	014556
C	TO THE P-FRAME	014557
C		014558
C		014559
C	-----CALL READIM (IRGFLX, 1, NB, 1, NBMAX)	014560
C		014561
C	VECTOR SIZE 1 BY NB TO DEFINE NUMBER OF ELASTIC MODES	014562
C		014563
C	FOR THE JTH BODY -	014564
C		014565
C	IRGFLX(J) = 0 - RIGID BODY	014566
C	IRGFLX(J) = N - FLEXIBLE BODY WITH N MODES	014567
C		014568
C		014569
C	-----CALL READIM (IFTSMW, 1, NSPT, 1, NSPMAX)	014570
C		014571
C	VECTOR SIZE 1 BY NSPT TO DEFINE SENSOR POINT LOCATIONS	014572
C		014573
C	FOR THE JTH SENSOR POINT -	014574
C		014575
C	IFTSMW(J) = BODY NUMBER FOR SENSOR POINT J	014576
C		014577
C	AT LEAST ONE SENSOR POINT MUST BE DEFINED IN	014578
C	THE SIMULATION MODEL	014579
C		014580
C		014581
C		014582
C		014583
C		014584
C	*** SUMMARY OF EULER ROTATION TYPES ***	014585
C		014586
C		014587
C	ITYPE PERMUTATION ORDER	014588
C		014589
C	1 (1,2,3)	014590
C		014591
C	2 (1,2,1)	014592
C		014593
C	3 (1,3,1)	014594
C		014595
C	4 (1,3,2)	014596
C		014597

C	5	(2,3,1)	014598
C			014599
C	6	(2,3,2)	014600
C			014601
C	7	(2,1,2)	014602
C			014603
C	8	(2,1,3)	014604
C			014605
C	9	(3,1,2)	014606
C			014607
C	10	(3,1,3)	014608
C			014609
C	11	(3,2,3)	014610
C			014611
C	12	(3,2,1)	014612
C			014613
C			014614
C	C-----CALL READIN (IHDATA, 7, NH, 7, NHMAX)		014615
C			014616
C	MATRIX SIZE 7 BY NH TO DEFINE HINGE POINT		014617
C	CONSTRAINT DATA		014618
C			014619
C	FOR THE JTH HINGE -		014620
C			014621
C	IHDATA(1,J) = EULER ROTATION TYPE TO ORIENT Q-TRIAD		014622
C	WRT TO P-TRIAD AT HINGE J (ITYPE)		014623
C	IHDATA(2,J) = HINGE CONSTRAINT TYPE - THETA 1 ROTATION		014624
C	IHDATA(3,J) = HINGE CONSTRAINT TYPE - THETA 2 ROTATION		014625
C	IHDATA(4,J) = HINGE CONSTRAINT TYPE - THETA 3 ROTATION		014626
C	IHDATA(5,J) = HINGE CONSTRAINT TYPE - X TRANSLATION		014627
C	IHDATA(6,J) = HINGE CONSTRAINT TYPE - Y TRANSLATION		014628
C	IHDATA(7,J) = HINGE CONSTRAINT TYPE - Z TRANSLATION		014629
C			014630
C	ROTATIONS REFER TO ORDER DEFINED BY IITYPE		014631
C			014632
C	TRANSLATIONS REFER TO P-TRIAD		014633
C			014634
C	IHDATA(2-7,J) = 0 - NO CONSTRAINT		014635
C	IHDATA(2-7,J) = 1 - FIXED CONSTRAINT		014636
C	IHDATA(2-7,J) = 2 - RHEGNOMIC CONSTRAINT		014637
C			014638
C			014639
C			014640
C			014641
C			014642
C			014643
C			014644
C	NOTE -- NUMBER OF BETA STATE VARIABLES COMPUTED FROM		014645
C	IHDATA AS SUM OF NUMBER OF ZEROS + SUM OF		014646
C	NUMBER OF TWOS IN ROWS 2 THRU 7		014647
C			014648
C	NUMBER OF CONSTRAINTS COMPUTED FROM		014649
C	IHDATA AS SUM OF NUMBER OF ONES + SUM OF		014650
C	NUMBER OF TWOS IN ROWS 2 THRU 7		014651
C			014652
C			014653
C	C-----CALL READ (BETAH, 6, NH, 6, NHMAX)		014654
C			014655
C	MATRIX SIZE 6 BY NH TO DEFINE INITIAL VALUES		014656
C	OF BETA WHICH ORIENT TWO BODIES ASSOCIATED		014657

C	WITH EACH HINGE	014658
C		014659
C	FOR THE JTH HINGE -	014660
C		014661
C	BETAH(1,J) = THETA 1 ROTATION	014662
C	BETAH(2,J) = THETA 2 ROTATION	014663
C	BETAH(3,J) = THETA 3 ROTATION	014664
C	BETAH(4,J) = X TRANSLATION	014665
C	BETAH(5,J) = Y TRANSLATION	014666
C	BETAH(6,J) = Z TRANSLATION	014667
C		014668
C		014669
C	-----CALL READ (BETAH, 6, NH, 6, NHMAX)	014670
C		014671
C	MATRIX SIZE 6 BY NH TO DEFINE INITIAL VALUES	014672
C	OF BETA DOT - TIME DERIVATIVE OF BETAH	014673
C	DESCRIBED PREVIOUSLY	014674
C		014675
C	FOR THE JTH HINGE -	014676
C		014677
C	BETAHD(1,J) = THETA 1 ROTATION RATE	014678
C	BETAHD(2,J) = THETA 2 ROTATION RATE	014679
C	BETAHD(3,J) = THETA 3 ROTATION RATE	014680
C	BETAHD(4,J) = X TRANSLATION RATE	014681
C	BETAHD(5,J) = Y TRANSLATION RATE	014682
C	BETAHD(6,J) = Z TRANSLATION RATE	014683
C		014684
C		014685
C	NOTE -- IF THE CORRESPONDING CONSTRAINT TYPE	014686
C	IS 1 OR 2, THE INITIAL BETAHD(1-6,J),	014687
C	INPUT HERE, WILL BE IGNORED AND SET TO	014688
C	ZERO OR TO THE RHEONCMICALLY PRESCRIBED	014689
C	VALUE, RESPECTIVELY	014690
C		014691
C		014692
C		014693
C		014694
C		014695
C		014696
C		014697
C		014698
C	NCFMC = NUMBER OF MOMENTUM WHEELS	014699
C		014700
C	IF (NCFMC .EQ. 0) GO TO 41	014701
C		014702
C		014703
C	-----CALL READIM (IMD, 3, NCFMC, 3, NMWMAX)	014704
C		014705
C	MATRIX SIZE 3 BY NCFMC TO DEFINE BASIC	014706
C	MOMENTUM WHEEL DATA	014707
C		014708
C	FOR THE JTH MOMENTUM WHEEL -	014709
C		014710
C	IMD(1,J) = SENSOR POINT NUMBER FOR THE WHEEL	014711
C	IMD(2,J) = SPIN AXIS NUMBER FOR THE WHEEL (1,2 OR 3)	014712
C	IMD(3,J) = 1 ACTIVE WHEEL	014713
C	= 0 CONSTANT SPEED WHEEL	014714
C		014715
C		014716
C	-----CALL READ (AMD, 2, NCFMC, 2, NMWMAX)	014717

C		014718
C	MATRIX SIZE 2 BY NOFMO TO DEFINE MOMENTUM	014719
C	WHEEL DATA	014720
C		014721
C	FOR THE JTH MOMENTUM WHEEL -	014722
C		014723
C	AMU(1,J) = INITIAL WHEEL SPIN RATE	014724
C	AMU(2,J) = SPIN INERTIA	014725
C		014726
C	41 CONTINUE	014727
C		014728
C		014729
C	-----CALL READ (TMDATA, 1, 3, 1, 3)	014730
C		014731
C	VECTOR SIZE 1 BY 3 CONTAINING TIME HISTORY	014732
C	INTEGRATION CONTROLS	014733
C		014734
C	TMDATA(1) = INITIAL TIME (START)	014735
C	TMDATA(2) = TIME INTERVAL (DELTA)	014736
C	TMDATA(3) = TERMINATION TIME (END)	014737
C		014738
C		014739
C		014740
C		014741
C		014742
C		014743
C		014744
C	-----READIM (IPDATA, 1, 3, 1, 3)	014745
C		014746
C	VECTOR SIZE 1 BY 3 CONTAINING INTEGER	014747
C	CONTROL DATA	014748
C		014749
C	IPDATA(1) = PRINT CONTROL FOR TIME RESPONSE - PRINT	014750
C	EVERY IPDATA(1) MULTIPLES OF DELTA	014751
C	IPDATA(2) = PLOT CONTROL FOR TIME RESPONSE - SAVE	014752
C	EVERY IPDATA(2) MULTIPLES OF DELTA	014753
C	IPDATA(3) = 0 PERFORM NONLINEAR TIME RESPONSE	014754
C	= 1 PERFORM LINEAR TIME OR FREQUENCY	014755
C	RESPONSE	014756
C		014757
C		014758
C	-----CALL READ (CNTDIA, 1, NONPAR, 1, KCON)	014759
C		014760
C	VECTOR SIZE 1 BY NONPAR FOR CONTROL SYSTEM VARIABLES	014761
C	AND USER SUPPLIED VARIABLES	014762
C		014763
C	THIS IS A CATCH-ALL VECTOR FOR USER PAK DATA	014764
C		014765
C	INFORMATION IS PUT INTO COMMON /CONTROL/ AND IS	014766
C	IDENTIFIED BY USER SUPPLIED EQUIVALENCE MAP	014767
C	IN USER PAK	014768
C		014769
C	FIRST DELTA ELEMENTS MUST CONTAIN INITIAL VALUES	014770
C	FOR DELTA VARIABLES IN STATE VECTOR	014771
C		014772
C	ADDITIONAL SPACE IS AVAILABLE TO THE USER	014773
C		014774
C		014775
C	-----CALL READ (WV, 1, 4, 1, 5)	014776
C		014777

C	VECTOR SIZE 1 BY 4 FOR GRAVITY GRADIENT DATA	014778
C		014779
C	WV(1) = PROJECTION OF GRAVITY VECTOR ON X INERTIAL AXIS	014780
C	WV(2) = PROJECTION OF GRAVITY VECTOR ON Y INERTIAL AXIS	014781
C	WV(3) = PROJECTION OF GRAVITY VECTOR ON Z INERTIAL AXIS	014782
C	WV(4) = RADIUS VECTOR FROM GRAVITY SOURCE TO GENERAL	014783
C	VICINITY OF BODY CLUSTER	014784
C		014785
C		014786
C	NOTE -- IF(WV(1)**2 + WV(2)**2 + WV(3)**2)	014787
C	EQ 0, WV(4) IS IGNORED	014788
C	NE 0, WV(4) MUST BE GT 1	014789
C		014790
C	(GRAVITY VECTOR COMPONENTS IN	014791
C	UNITS OF ACCELERATION)	014792
C		014793
C		014794
C		014795
C		014796
C		014797
C		014798
C		014799
C	NB = NUMBER OF BODIES	014800
C		014801
C	DO 20 N=1,NB	014802
C		014803
C	IRGFLX(N) EQ 0 - RIGID BODY	014804
C	IRGFLX(N) GT 0 - FLEXIBLE BODY	014805
C		014806
C	IF (IRGFLX(N) .EQ. 0) GO TO 25	014807
C		014808
C	-----READ(NIT,FORMAT = 15) NTYPE	014809
C		014810
C	NTYPE = 1 - LUMPED MASS REPRESENTATION	014811
C	NTYPE = 2 - CONSISTENT MASS REPRESENTATION	014812
C		014813
C	IF (NTYPE .EQ. 1) CALL MSMODL(N) --- SEE FOLLOWING	014814
C		014815
C	IF (NTYPE .EQ. 2) CALL MSMCDC(N) --- SEE FOLLOWING	014816
C		014817
C	GO TO 20	014818
C		014819
C	25 CALL MRIGID(N) --- SEE FOLLOWING	014820
C		014821
C	20 CONTINUE	014822
C		014823
C	RETURN	014824
C	END	014825
C		014826
C		014827
C		014828
C		014829
C		014830
C	*****	014831
C	C*	014832
C	C* SUBROUTINE MRIGID - INPUT FOR RIGID BODY	014833
C	C*	014834
C	*****	014835
C		014836
C		014837

C-----CALL READ (V, 1, 4, 1, 0)	C14838
C	C14839
C	C14840
C	C14841
C	C14842
C	C14843
C	C14844
C	C14845
C	C14846
C	C14847
C-----CALL READ (V, 1, 6, 1, 0)	C14848
C	C14849
C	C14850
C	C14851
C	C14852
C	C14853
C	C14854
C	C14855
C	C14856
C	C14857
C	C14858
C	C14859
C	C14860
C	C14861
C	C14862
C	C14863
C	C14864
C	C14865
C	C14866
C	C14867
C	C14868
C	C14869
C	C14870
C	C14871
C-----READ(NIT,FORMAT = 215) NCH, ITYPE	C14872
C	C14873
C	C14874
C	C14875
C	C14876
C	C14877
C	C14878
C	C14879
C	C14880
C	C14881
C	C14882
C	C14883
C-----READ(NIT,FORMAT = 3010.3) (V(J),J=1,3)	C14884
C	C14885
C	C14886
C	C14887
C	C14888
C	C14889
C	C14890
C	C14891
C	C14892
C	C14893
C-----READ(NIT,FORMAT = 3010.3) (DH(J),J=1,3)	C14894
C	C14895
C	C14896
C	C14897

C	DH(1) = X (BODY REF POINT TO HINGE POINT, BODY TRIAD)	014898
C	DH(2) = Y (BODY REF POINT TO HINGE POINT, BODY TRIAD)	014899
C	DH(3) = Z (BODY REF POINT TO HINGE POINT, BODY TRIAD)	014900
C		014901
C	10 CONTINUE	014902
C		014903
C	NSB = NUMBER OF SENSOR POINTS ON BODY N	014904
C		014905
C	IF (NSB .EQ. 0) RETURN	014906
C		014907
C	DO 20 I=1,NSB	014908
C		014909
C		014910
C	-----READ(NIT,FORMAT = 215) NCS, ITYPE	014911
C		014912
C	NCS = SENSOR POINT NUMBER	014913
C	ITYPE = EULER ROTATION TYPE TO ORIENT SENSOR POINT	014914
C	TRIAD WRT BODY TRIAD	014915
C		014916
C		014917
C	-----READ(NIT,FORMAT = 3010.3) (V(J),J=1,3)	014918
C		014919
C	EULER ANGLES TO ORIENT SENSOR POINT TRIAD - PERMUTATION	014920
C	ORDER DEFINED BY ITYPE	014921
C		014922
C	V(1) = THETA 1 (FIRST ROTATION)	014923
C	V(2) = THETA 2 (SECND ROTATION)	014924
C	V(3) = THETA 3 (THIRD ROTATION)	014925
C		014926
C		014927
C		014928
C		014929
C		014930
C		014931
C		014932
C		014933
C	-----READ(NIT,FORMAT = 3010.3) (DS(J),J=1,3)	014934
C		014935
C	VECTOR TO POSITION SENSOR POINT TRIAD WRT BODY TRIAD	014936
C		014937
C	DS(1) = X (BODY REF POINT TO SENSOR POINT, BODY TRIAD)	014938
C	DS(2) = Y (BODY REF POINT TO SENSOR POINT, BODY TRIAD)	014939
C	DS(3) = Z (BODY REF POINT TO SENSOR POINT, BODY TRIAD)	014940
C		014941
C	20 CONTINUE	014942
C		014943
C	RETURN	014944
C	END	014945
C		014946
C		014947
C		014948
C		014949
C		014950
C	*****	014951
C	C*	014952
C	C* SUBROUTINE NSMODL - INPUT FOR FLEXIBLE BODY, LUMPED MASS MATRIX	014953
C	C*	014954
C	*****	014955
C		014956
C		014957

```

C-----CALL READ (A, NJ, 1, KJCINT, KMCDE)
C
C          MATRIX SIZE NJ BY 1 WHERE NJ = NUMBER OF JOINTS
C          ON BODY N
C
C          FOR THE ITH JOINT -
C
C          A(I,1) = JOINT MASS
C
C-----CALL READ (A, NJ, 6, KJCINT, KMCDE)
C
C          MATRIX SIZE NJ BY 6
C
C          FOR THE ITH JOINT -
C
C          A(I,1) = JOINT INERTIA, JXX
C          A(I,2) = JOINT INERTIA, JYY
C          A(I,3) = JOINT INERTIA, JZZ
C          A(I,4) = JOINT INERTIA, JXY
C          A(I,5) = JOINT INERTIA, JXZ
C          A(I,6) = JOINT INERTIA, JYZ
C
C                      JXX -JXY -JXZ
C          INERTIA MATRIX = -JXY JYY -JYZ
C                      -JXZ -JYZ JZZ
C          NOTE -- J(XY) = VOL INTEGRAL OF (XY DM), ETC.
C
C          DO 5 K=1,2
C
C-----CALL READ (A, NJ, 3, KJCINT, KMCDE)
C
C          MATRIX SIZE NJ BY 3
C
C          FOR THE ITH JOINT -
C
C          K=1  A(I,1) = JOINT STATIC MASS MOMENT, SX
C              A(I,2) = JOINT STATIC MASS MOMENT, SY
C              A(I,3) = JOINT STATIC MASS MOMENT, SZ
C
C          K=2  A(I,1) = X (BODY REF POINT TO JOINT, BODY TRIAD)
C              A(I,2) = Y (BODY REF POINT TO JOINT, BODY TRIAD)
C              A(I,3) = Z (BODY REF POINT TO JOINT, BODY TRIAD)
C
C          5 CONTINUE
C
C
C          M      0      0      0      SZ -SY
C          J      0      0      -SZ      0      SX
C          0      0      0      M      SY -SX      0
C          JOINT INERTIA MATRIX =  0 -SZ      SY      JXX -JXY -JXZ
C                                SZ      0 -SX -JXY      JYY -JYZ
C                                -SY      SX      0 -JXZ -JYZ      JZZ
C
C
C
C
C
C
C
C
C
C
C

```

```

014958
014959
014960
014961
014962
014963
014964
014965
014966
014967
014968
014969
014970
014971
014972
014973
014974
014975
014976
014977
014978
014979
014980
014981
014982
014983
014984
014985
014986
014987
014988
014989
014990
014991
014992
014993
014994
014995
014996
014997
014998
014999
015000
015001
015002
015003
015004
015005
015006
015007
015008
015009
015010
015011
015012
015013
015014
015015
015016
015017

```

C		015018
C		015019
C	DO 10 K=1,6	015020
C		015021
C		015022
C	-----CALL READ (A, NJ, NE, KJOINT, KMODE)	015023
C		015024
C	MATRIX SIZE NJ BY NE WHERE NE = NUMBER OF ELASTIC	015025
C	MODES RETAINED FOR BODY N	015026
C		015027
C	FOR THE ITH JOINT -	015028
C		015029
C	K=1 A(I,J) = X DISPLACEMENT AT JOINT, MODE J	015030
C	K=2 A(I,J) = Y DISPLACEMENT AT JOINT, MODE J	015031
C	K=3 A(I,J) = Z DISPLACEMENT AT JOINT, MODE J	015032
C	K=4 A(I,J) = THETA X ROTATION AT JOINT, MODE J	015033
C	K=5 A(I,J) = THETA Y ROTATION AT JOINT, MODE J	015034
C	K=6 A(I,J) = THETA Z ROTATION AT JOINT, MODE J	015035
C		015036
C	10 CONTINUE	015037
C		015038
C		015039
C	DO 20 K=1,2	015040
C		015041
C		015042
C	-----CALL READ (A, NE, NE, KJOINT, KMODE)	015043
C		015044
C	MATRIX SIZE NE BY NE	015045
C		015046
C	K=1 A = MODAL STIFFNESS	015047
C		015048
C	K=2 A = MODAL DAMPING	015049
C		015050
C	20 CONTINUE	015051
C		015052
C		015053
C	-----READ(NIT,FORMAT = 6D10.3) (A(J),J=1,NE)	015054
C		015055
C	VECTOR OF INITIAL MODAL DEFLECTION COORDINATES	015056
C		015057
C	-----READ(NIT,FORMAT = 6D10.3) (A(J),J=1,NE)	015058
C		015059
C	VECTOR OF INITIAL MODAL VELOCITY COORDINATES	015060
C		015061
C		015062
C		015063
C		015064
C		015065
C		015066
C		015067
C		015068
C	NOTE -- FOLLOWING EULER ANGLES MEASURED	015069
C	IN UNDEFORMED CONFIGURATION	015070
C		015071
C		015072
C	NHB = NUMBER OF HINGES ON BODY N - EXCLUSIVE OF HINGE 1,	015073
C	BODY 1	015074
C		015075
C	DO 100 L=1,NHB	015076
C		015077

C		C15078
C-----READ(NIT,FORMAT = 315) NHI, ITYPE, JOINT		C15079
C		C15080
C	NHI = HINGE NUMBER	C15081
C	ITYPE = EULER ROTATION TYPE TO ORIENT HINGE	C15082
C	TRIAO = BODY TRIAD	C15083
C	JOINT = JOINT NUMBER CORRESPONDING TO HINGE POINT	C15084
C		C15085
C		C15086
C-----READ(NIT,FORMAT = 3010.3) (WV(J),J=1,3)		C15087
C		C15088
C	EULER ANGLES TO ORIENT HINGE TRIAD - PERMUTATION	C15089
C	ORDER DEFINED BY ITYPE	C15090
C		C15091
C	WV(1) = THETA 1 (FIRST ROTATION)	C15092
C	WV(2) = THETA 2 (SECOND ROTATION)	C15093
C	WV(3) = THETA 3 (THIRD ROTATION)	C15094
C		C15095
C	150 CONTINUE	C15096
C		C15097
C		C15098
C	NSO = NUMBER OF SENSOR POINTS IN BODY B	C15099
C		C15100
C	IF (NSO .EQ. 0) RETURN	C15101
C		C15102
C	LD 100 LE1,NSO	C15103
C		C15104
C		C15105
C-----READ(NIT,FORMAT = 315) NOS, ITYPE, JOINT		C15106
C		C15107
C	NOS = SENSOR POINT NUMBER	C15108
C	ITYPE = EULER ROTATION TYPE TO ORIENT SENSOR POINT	C15109
C	TRIAO = BODY TRIAD	C15110
C	JOINT = JOINT NUMBER CORRESPONDING TO SENSOR POINT	C15111
C		C15112
C		C15113
C		C15114
C		C15115
C		C15116
C		C15117
C		C15118
C		C15119
C-----READ(NIT,FORMAT = 3010.3) (WV(J),J=1,3)		C15120
C		C15121
C	EULER ANGLES TO ORIENT SENSOR POINT TRIAD - PERMUTATION	C15122
C	ORDER DEFINED BY ITYPE	C15123
C		C15124
C	WV(1) = THETA 1 (FIRST ROTATION)	C15125
C	WV(2) = THETA 2 (SECOND ROTATION)	C15126
C	WV(3) = THETA 3 (THIRD ROTATION)	C15127
C		C15128
C	100 CONTINUE	C15129
C		C15130
C	RETURN	C15131
C	END	C15132
C		C15133
C		C15134
C		C15135
C		C15136
C		C15137


```

C***** 015138
C* 015139
C* SUBROUTINE MSMODE = INPUT FOR FLEXIBLE BODY, CONSISTENT MASS MATRIX . 015140
C* 015141
C***** 015142
C 015143
C 015144
C-----READ(NIT,FORMAT = 315) IFRBM, IDIAK, IDIAD 015145
C 015146
C          IFRBM = RIGID BODY MODE CONTROL VARIABLE 015147
C 015148
C          = 0 WHEN RIGID BODY MODES CAN BE 015149
C          OPERATED ON VIA ROW-COL 015150
C          INTERCHANGES TO OBTAIN A RIGID 015151
C          BODY MODAL MATRIX OF THE FORM- 015152
C 015153
C          ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** 015154
C          * HX * * 1 0 0 0 Z -Y * * QX * 015155
C          * * * * * * * * * * 015156
C          * HY * * 0 1 0 -Z 0 X * * QY * 015157
C          * * * * * * * * * * 015158
C          * HZ * * 0 0 1 Y -X 0 * * QZ * 015159
C          * * = * * * * * * * * 015160
C          * IX * * 0 0 0 1 0 0 * * QTX * 015161
C          * * * * * * * * * * 015162
C          * IY * * 0 0 0 0 1 0 * * QTY * 015163
C          * * * * * * * * * * 015164
C          * IZ * * 0 0 0 0 0 1 * * QTZ * 015165
C          ** ** ** * * * * * 015166
C 015167
C 015168
C          = 1 WHEN ABOVE IS NOT TRUE, SUCH 015169
C          AS FOR AN ORTHONORMAL SET OF 015170
C          RIGID BODY VECTORS. DATA WILL 015171
C          BE READ IN LATER AND THE ABOVE 015172
C          FORM WILL BE CREATED WITHIN 015173
C          THIS SUBROUTINE 015174
C 015175
C 015176
C 015177
C 015178
C 015179
C 015180
C 015181
C          IDIAK = 0 IF MODAL STIFFNESS MATRIX 015182
C          IS NOT DIAGONAL 015183
C 015184
C          = 1 IF MODAL STIFFNESS MATRIX 015185
C          IS DIAGONAL 015186
C 015187
C 015188
C          IDIAD = 0 IF MODAL DAMPING MATRIX 015189
C          IS NOT DIAGONAL 015190
C 015191
C          = 1 IF MODAL DAMPING MATRIX 015192
C          IS DIAGONAL 015193
C 015194
C 015195
C-----CALL READIM (JDOF, Nx, N0, KJDOF, 6) 015196
C 015197

```

C	MATRIX SIZE NX BY 6 DEFINING JOINT DEGREES	015198
C	OF FREEDOM FOR NX JOINTS ON BODY N	015199
C		015200
C	FOR THE ITH JOINT -	015201
C		015202
C	JDOF(1,1) = X DEG. OF FREEDOM FOR POINT 1.	015203
C	JDOF(1,2) = Y DEG. OF FREEDOM FOR POINT 1.	015204
C	JDOF(1,3) = Z DEG. OF FREEDOM FOR POINT 1.	015205
C	JDOF(1,4) = TX DEG. OF FREEDOM FOR POINT 1.	015206
C	JDOF(1,5) = TY DEG. OF FREEDOM FOR POINT 1.	015207
C	JDOF(1,6) = TZ DEG. OF FREEDOM FOR POINT 1.	015208
C		015209
C	POINT 1 IS CONSISTANT WITH THE	015210
C	MODAL MATRIX ROW 1. BEFORE REORDING.	015211
C		015212
C	JDOF WILL BE INTERROGATED AND THE	015213
C	COORDINATES WILL BE ORDERED AS --	015214
C		015215
C	HAX(1) FOR JOINTS 1 THRU NX	015216
C	HAY(1) FOR JOINTS 1 THRU NX	015217
C	HAZ(1) FOR JOINTS 1 THRU NX	015218
C	HAX(1) FOR JOINTS 1 THRU NX	015219
C	HAY(1) FOR JOINTS 1 THRU NX	015220
C	HAZ(1) FOR JOINTS 1 THRU NX	015221
C		015222
C		015223
C		015224
C		015225
C		015226
C		015227
C		015228
C	-----CALL READIM (JV, 1, NMOT, 1, KAB)	015229
C		015230
C	VECTOR SIZE 1 BY NMOT = NUMBER OF RIGID BODY MODES +	015231
C	NUMBER OF ELASTIC MODES	015232
C		015233
C	JV(J) = IND WHERE IND IS COLUMN NO. WHICH COL(J) OF	015234
C	ORIGINAL MODAL MATRIX WILL APPEAR IN	015235
C	REVISED MODE MATRIX	015236
C	IND GT 0 REPLACE COLUMN	015237
C	IND EQ 0 DELETE COLUMN	015238
C	IND LT 0 REPLACE COLUMN, CHANGE SIGNS	015239
C		015240
C		015241
C	-----CALL READ (A, NRA, NCA, KAB, KAB)	015242
C		015243
C	MATRIX SIZE C*NX BY C*NX CONTAINS CONSISTENT	015244
C	MASS REPRESENTATION	015245
C		015246
C	ROW-COLUMN COORDINATE ORDER MUST BE CONSISTENT	015247
C	WITH THE DEGREE OF FREEDOM TABLE, JDOF	015248
C		015249
C		015250
C	-----CALL READ (A, NRA, NMOT, KAB, KAB)	015251
C		015252
C	MATRIX SIZE C*NX BY NMOT CONTAINS MODAL DEFINITION	015253
C		015254
C	THE COLUMNS (MODE ORDER) WILL BE REORDERED BY	015255
C	THE INPUT VECTOR JV	015256
C		015257

C		015258
C	IF (IDIAK .EQ. 0 .AND. IDIAD .EQ. 0) GO TO 11	015259
C		015260
C	-----CALL READ (OM2, 1, NMCDT, 1, KAB)	015261
C		015262
C	VECTOR SIZE 1'BY NMCDT CONTAINING SQUARES OF NATURAL	015263
C	FREQUENCIES	015264
C		015265
C	OM2(J) = SQUARE OF JTH NATURAL FREQUENCY CORRESPONDING	015266
C	TO JTH INPUT MODE SHAPE	015267
C		015268
C	11 CONTINUE	015269
C		015270
C		015271
C		015272
C		015273
C		015274
C		015275
C		015276
C	IF (IFRBM .EQ. 0) GO TO 5	015277
C		015278
C		015279
C	-----READ(NIT,FORMAT = 15) JTYPCL	015280
C		015281
C	JTYPCL = REFERENCE JOINT NUMBER WHOSE GEOMETRIC	015282
C	POSITION COORDINATES WILL BE USED TO	015283
C	ESTABLISH RIGID BODY MODAL MATRIX	015284
C		015285
C	-----READ(NIT,FORMAT = 3010.3) (CM2(J),J=1,3)	015286
C		015287
C	OM2(1) = X COMPONENT OF VECTOR THAT LOCATES JTYPCL PT	015288
C	OM2(2) = Y COMPONENT OF VECTOR THAT LOCATES JTYPCL PT	015289
C	OM2(3) = Z COMPONENT OF VECTOR THAT LOCATES JTYPCL PT	015290
C		015291
C	VECTOR COMPONENTS REFERED TO BODY TRIAD	015292
C		015293
C	5 CONTINUE	015294
C		015295
C		015296
C	IF (IDIAK .EQ. 1) GO TO 50	015297
C		015298
C	-----CALL READ (A, NRA, NCA, KAB, KAB)	015299
C		015300
C	STIFFNESS MATRIX -- SEE NOTE BELOW	015301
C		015302
C	50 CONTINUE	015303
C		015304
C	IF (IDIAD .EQ. 1) GO TO 60	015305
C		015306
C		015307
C	-----CALL READ (A, NRA, NCA, KAB, KAB)	015308
C		015309
C	DAMPING MATRIX -- SEE NOTE BELOW	015310
C		015311
C	GO TO 61	015312
C		015313
C	60 CONTINUE	015314
C		015315
C		015316
C	-----READ(NIT,FORMAT = 6010.3) (CM2(J),J=1,NR)	015317

C		C15318
C	VECTOR SIZE 1 BY NE = NUMBER OF ELASTIC MODES	C15319
C	RETAINED VIA INPUT JV SELECTION VECTOR	C15320
C		C15321
C	CM2(J) = MODAL DAMPING RATIO FOR JTH ELASTIC MODE	C15322
C		C15323
C	GO CONTINUE	C15324
C		C15325
C		C15326
C		C15327
C		C15328
C		C15329
C		C15330
C		C15331
C	NOTE FOR STIFFNESS AND DAMPING MATRICES	C15332
C		C15333
C	COORDINATE ORDER ASSUMED CONSISTENT WITH	C15334
C	THE REORDERED COORDINATE DESCRIPTION	C15335
C	AFTER THE DEGREE OF FREEDOM TABLE (JDOP)	C15336
C	HAS BEEN INTERROGATED AND THE FOLLOWING	C15337
C	ORDER ESTABLISHED	C15338
C		C15339
C		C15340
C	HX(I), I=1,2,...,N	C15341
C	.	C15342
C	.	C15343
C	HY(I), I=1,2,...,N	C15344
C	.	C15345
C	.	C15346
C	HZ(I), I=1,2,...,N	C15347
C	.	C15348
C	.	C15349
C	TX(I), I=1,2,...,N	C15350
C	.	C15351
C	.	C15352
C	TY(I), I=1,2,...,N	C15353
C	.	C15354
C	.	C15355
C	TZ(I), I=1,2,...,N	C15356
C	.	C15357
C	.	C15358
C		C15359
C		C15360
C	-----READ(NIT,FORMAT = 8D10.3) (CM2(J),J=1,NE)	C15361
C		C15362
C	VECTOR SIZE 1 BY NE CONTAINING INITIAL	C15363
C	MODAL DEFLECTION COORDINATES	C15364
C		C15365
C		C15366
C	-----READ(NIT,FORMAT = 8D10.3) (CM2(J),J=1,NE)	C15367
C		C15368
C	VECTOR SIZE 1 BY NE CONTAINING INITIAL	C15369
C	MODAL VELOCITY COORDINATES	C15370
C		C15371
C		C15372
C		C15373
C		C15374
C		C15375
C		C15376
C		C15377

C	NOTE -- FOLLOWING EULER ANGLES MEASURED	015378
C	IN UNDEFORMED CONFIGURATION	015379
C		015380
C		015381
C	NHB = NUMBER OF HINGES ON BODY N - EXCLUSIVE OF HINGE 1,	015382
C	BODY 1	015383
C		015384
C	DO 110 L=1,NHB	015385
C		015386
C		015387
C	-----READ(NIT,FORMAT = 315) NHH, ITYPE, JOINT	015388
C		015389
C	NHH = HINGE NUMBER	015390
C	ITYPE = EULER ROTATION TYPE TO ORIENT HINGE	015391
C	TRIAD WRT BODY TRIAD	015392
C	JOINT = JOINT NUMBER CORRESPONDING TO HINGE POINT	015393
C		015394
C		015395
C	-----READ(NIT,FORMAT = 3010.3) (OM2(J),J=1,3)	015396
C		015397
C	EULER ANGLES TO ORIENT HINGE TRIAD - PERMUTATION	015398
C	ORDER DEFINED BY ITYPE	015399
C		015400
C	OM2(1) = THETA 1 (FIRST ROTATION)	015401
C	OM2(2) = THETA 2 (SECND ROTATION)	015402
C	OM2(3) = THETA 3 (THIRD ROTATION)	015403
C		015404
C	110 CONTINUE	015405
C		015406
C		015407
C	NSB = NUMBER OF SENSOR POINTS ON BODY N	015408
C		015409
C	IF (NSB .EQ. 0) RETURN	015410
C		015411
C	DO 120 L=1,NSB	015412
C		015413
C		015414
C	-----READ(NIT,FORMAT = 315) NNS, ITYPE, JOINT	015415
C		015416
C	NNS = SENSOR POINT NUMBER	015417
C	ITYPE = EULER ROTATION TYPE TO ORIENT SENSOR POINT	015418
C	TRIAD WRT BODY TRIAD	015419
C	JOINT = JOINT NUMBER CORRESPONDING TO SENSOR POINT	015420
C		015421
C		015422
C		015423
C		015424
C		015425
C		015426
C		015427
C		015428
C		015429
C	-----READ(NIT,FORMAT = 3010.3) (OM2(J),J=1,3)	015430
C		015431
C	EULER ANGLES TO ORIENT SENSOR POINT TRIAD - PERMUTATION	015432
C	ORDER DEFINED BY ITYPE	015433
C		015434
C	OM2(1) = THETA 1 (FIRST ROTATION)	015435
C	OM2(2) = THETA 2 (SECND ROTATION)	015436
C	OM2(3) = THETA 3 (THIRD ROTATION)	015437

C		015438
C	120 CONTINUE	015439
C		015440
C	RETURN	015441
C	END	015442
C		015443
C		015444
C		015445
C		015446
C		015447
C	*****	015448
C	C*	015449
C	C* SUBROUTINE CYN520 - INPUT USER PAK DATA (IF ANY)	015450
C	C*	015451
C	*****	015452
C		015453
C		015454
C	THE ONLY DATA READ HERE ARE THOSE ASSOCIATED	015455
C	WITH POLYNOMIAL DEFINITION OF TRANSFER	015456
C	FUNCTIONS	015457
C		015458
C	DATA (IF ANY) IS READ ON THE FIRST CALL TO	015459
C	SUBROUTINE CYN520 WHERE THE INTEGER VARIABLE	015460
C	NPLY IS DEFINED BY A USER SUPPLIED DATA	015461
C	STATEMENT	015462
C		015463
C	NPLY = NUMBER OF SETS OF POLYNOMIAL TRANSFER	015464
C	FUNCTION COEFFICIENTS TO READ	015465
C		015466
C	IF (NPLY .EQ. 0) GO TO 10	015467
C		015468
C	DO 20 K=1,NPLY	015469
C		015470
C	K2 = 2*K-1	015471
C		015472
C		015473
C	-----CALL READ (CPLY(1,K2), KPLY(K), 2, KRY, KLY)	015474
C		015475
C	20 CONTINUE	015476
C		015477
C	MATRIX SIZE KPLY(K) BY 2 OF POLYNOMIAL COEFFICIENTS	015478
C		015479
C	CPLY(1,1) = DENOMINATOR COEFFICIENTS IN	015480
C	ASCENDING ORDER	015481
C	CPLY(1,2) = NUMERATOR COEFFICIENTS IN	015482
C	ASCENDING ORDER	015483
C		015484
C	NOTE -- KPLY(K) IS DEFINED AT INPUT (SUBROUTINE READ)	015485
C	AND IS ONE GREATER THAN THE ORDER OF THE	015486
C	DENOMINATOR POLYNOMIAL	015487
C		015488
C	ORDER OF NUMERATOR POLYNOMIAL MUST NOT EXCEED	015489
C	ORDER OF DENOMINATOR POLYNOMIAL	015490
C		015491
C	10 CONTINUE	015492
C		015493
C	RETURN	015494
C	END	015495
C		015496
C		015497

C	015498
C	015499
C	015500
C*****	015501
C*	015502
C* SUBROUTINE DYN30 - INPUT FOR TIME HISTORY PLOT OUTPUT	015503
C*	015504
C*****	015505
C	015506
C	015507
C-----READ(NIT,FORMAT = 10A8) (ICTITL(I),I=1,10)	015508
C	015509
C	015510
C	015511
C	015512
C	015513
C-----READ(NIT,FORMAT = 15) NSET	015514
C	015515
C	015516
C	015517
C	015518
C	015519
C	015520
C	015521
C	015522
C	015523
C	015524
C	015525
C	015526
C	015527
C-----READ(NIT,FORMAT = 15) JPL	015528
C	015529
C	015530
C	015531
C	015532
C	015533
C	015534
C	015535
C	015536
C	015537
C	015538
C	015539
C	015540
C	015541
C	015542
C	015543
C	015544
C	015545
C	015546
C	015547
C	015548
C	015549
C	015550
C	015551
C	015552
C	015553
C	015554
C	015555
C	015556
C	015557

C		015558
C		015559
C		015560
C		015561
C		015562
C	NOTE ---	015563
C		015564
C		015565
C		015566
C		015567
C		015568
C		015569
C		015570
C		015571
C		015572
C		015573
C		015574
C		015575
C		015576
C		015577
C		015578
C		015579
C		015580
C		015581
C		015582
C		015583
C		015584
C		015585
C		015586
C		015587
C		015588
C		015589
C		015590
C		015591
C		015592
C		015593
C		015594
C		015595
C		015596
C		015597
C		015598
C		015599
C		015600
C		015601
C		015602
C		015603
C		015604
C		015605
C		015606
C		015607
C		015608
C		015609
C		015610
C	NONLINEAR	015611
C	ANALYSIS	015612
C		015613
C		015614
C		015615
C		015616
C		015617

```

      NB
      ****
      *
      NEQ = * (IRGFLX(J) + NOJ + NDELTA + NDELTA
      *
      ****
      J=1

      NB          NOFNO
      ****        ****
      *           *
      NO = * IRGFLX(J) + * IHD(K) + C*NB
      *           *
      ****        ****
      J=1          K=1

      NDELTA = SUM OF NUMBER OF ZEROS + SUM OF
              NUMBER OF TWOS IN ROWS 2 THRU 7
              OF ARRAY IHDATA

      NLAM = SUM OF NUMBER OF ONES + SUM OF
            NUMBER OF TWOS IN ROWS 2 THRU 7
            OF ARRAY IHDATA

ORDER OF VARIABLES AND
SIZE FOR A SINGLE RECORD
(FOR A SINGLE TIME, T).

      VARIABLE ID.          SIZE
NONLINEAR ANALYSIS
      TIME                  1
      Y                     NEQ
      YDOT                  NEQ
      LAMBDA                 NLAM

```


C				015618
C		U	NU	015619
C				015620
C		HX, HY, HZ,		015621
C		PX, PY, PZ	U*NB	015622
C				015623
C		TOTAL ANGULAR		015624
C		MOMENTUM VECTOR		015625
C		COMPONENTS (X,Y,Z).	3	015626
C				015627
C		TOTAL LINEAR	INERTIAL	015628
C		MOMENTUM VECTOR	FRAME.	015629
C		COMPONENTS (X,Y,Z).	3	015630
C				015631
C		BODY KINETIC		015632
C		ENERGIES.	NB	015633
C				015634
C		BODY POTENTIAL		015635
C		ENERGIES.	NB	015636
C				015637
C		TOTAL ANGULAR MOM.,		015638
C		TOTAL LINEAR MOM.,		015639
C		TOTAL K.E.,		015640
C		TOTAL P.E.,		015641
C		TOTAL ENERGY	3	015642
C				015643
C				015644
C				015645
C	LINEAR			015646
C	ANALYSIS	TIME	1	015647
C				015648
C		Y	NEQ	015649
C		YDOT	NEQ	015650
C				015651
C				015652
C				015653
C				015654
C				015655
C				015656
C				015657
C	-----	READ(NIT,FORMAT = 1615) (JVPL(J),J=1,JPL)		015658
C				015659
C				015660
C		JVPL(J) = INTEGER DENOTING GLOBAL		015661
C		LOCATION OF JTH SELECTED		015662
C		VARIABLE FROM THE NCPLT		015663
C		LONG ARRAY.		015664
C				015665
C	20	CONTINUE		015666
C				015667
C				015668
C	-----	READ(NIT,FORMAT = 515) NCI, (NCD(I),I=1,3), NGRID		015669
C				015670
C		NCI = ELEMENT LOCATION (LOCAL WRT		015671
C		JVPL ARRAY) FOR THE INDEPEN-		015672
C		DENT PLOT VARIABLE.		015673
C				015674
C		NCD = ELEMENT LOCATION (LOCAL WRT		015675
C		JVPL ARRAY) FOR UP TO 3		015676
C		DEPENDENT VARIABLES TO PLOT		015677

C		015678
C	SIMULTANECUSLY VERSES THE	015679
C	NCI DEPENDENT VARIABLE.	015680
C		015681
C	NGRID = NO. OF PLOT FRAMES TO USE	015682
C	FOR PLOTTING THE NCI-NCU	015683
C	GROUP. IE. THE NO. OF FRAMES	015684
C	TO USE SIDE BY SIDE TO	015685
C	EXHAUST THE RANGE OF THE	015686
C	INDEPENDENT VARIABLE.	015687
C		015688
C	IF (NCI .EQ. 0) GO TO 1000	015689
C		015690
C	THIS IS ONE TO PROCEED TO NEXT SET	015691
C		015692
C	THIS OPEN ENDED LOOP PERMITS MANY SELECTIONS OF THE	015693
C	JVPL DATA VARIABLES WITH REGARD TO INDEPENDENT AND	015694
C	DEPENDENT VARIABLES IN ORDER TO FORM CROSS-PLOTS.	015695
C		015696
C		015697
C		015698
C		015699
C		015700
C		015701
C		015702
C		015703
C	-----READ(NIT,FORMAT = A6,2X,A8,2X,6A8) TITL1, TITL2,	015704
C	* (PTITL(1),I=1,6)	015705
C		015706
C	ALPHANUMERIC TITLING INFORMATION	015707
C	TO INCLUDE ON PLOTTED OUTPUT.	015708
C		015709
C	TITL1 = INDEPENDENT VARIABLE TITLE.	015710
C		015711
C	TITL2 = DEPENDENT VARIABLE TITLE.	015712
C		015713
C	PTITL = OVERALL TITLE FOR PARTICULAR	015714
C	IDENTIFICATION OF THIS FRAME.	015715
C		015716
C		015717
C	GO TO 20	015718
C		015719
C	1000 CONTINUE	015720
C		015721
C	RETURN	015722
C	END	015723
C		015724
C		015725
C		015726
C		015727
C		015728
C	C*****	015729
C	C*	015730
C	C* SUBROUTINE DYN540 - INPUT FOR LINEARIZED SYSTEM ANALYSES	015731
C	C*	015732
C	C*****	015733
C		015734
C		015735
C	-----READ(NIT,FORMAT = A4) LNAM	015736
C		015737

C	LNAM = 4HTIME FOR LINEARIZED	015738
C	TIME RESPONSE	015739
C		015740
C	= 4HFREQ FOR LINEARIZED	015741
C	FREQUENCY DOMAIN STUDY	015742
C		015743
C	IF (LNAM .EQ. 4HTIME) RETURN	015744
C		015745
C	IF (LNAM .EQ. 4H) RETURN	015746
C		015747
C		015748
C	-----CALL READIM (LRY, 9, NCYC, 9, KR)	015749
C		015750
C	MATRIX SIZE 9 BY NCYC DEFINING FREQUENCY	015751
C	ANALYSIS CONTROL DATA	015752
C		015753
C	NCYC = NO. OF SEPARATE TRANSFER	015754
C	FUNCTION CYCLES TO CONSIDER	015755
C	FOR THIS SIMULATION	015756
C		015757
C	FOR THE JTH CYCLE -	015758
C		015759
C	LRY(1,J) = ITYPE --- INTEGER CLASSIFYING THE	015760
C	TRANSFER FUNCTION TYPE.	015761
C		015762
C	ITYPE = 1 PLANT ONLY (S)	015763
C	= 2 CONTROLLER (H)	015764
C	= 3 OPEN LOOP (GH)	015765
C	= 4 OPEN LOOP (HG)	015766
C	= 5 CLOSED LOOP (GH/(1+GH))	015767
C	= 6 CLOSED LOOP	015768
C	= 7 PSEUDO OPEN LOOP,	015769
C	= 8 SPECIAL CASE (SEE BELOW)	015770
C	PERMITS OPENING OF A	015771
C	SINGLE RETURN LOOP.	015772
C		015773
C	A MINUS SIGN ON ITYPE	015774
C	EQUAL (3,4,5,OR 7) INDICATES	015775
C	NEGATIVE CONTROLLER FEEDBACK.	015776
C	A MINUS SIGN ON ITYPE EQUAL 8	015777
C	INDICATES COMPLETELY CLOSED	015778
C	LOOP TRANSFER FUNCTION.	015779
C		015780
C		015781
C		015782
C		015783
C		015784
C	LRY(2,J) = ITPIN --- TRANSFER FUNCTION INPUT	015785
C	VARIABLE IDENTIFICATION.	015786
C	THIS INTEGER IS A LOCAL	015787
C	IDENTIFICATION INTEGER.	015788
C	REFERENCING (DEPENDING	015789
C	UPON ITYPE) EITHER A	015790
C	SENSOR SIGNAL OR A	015791
C	CONTROLLER OUTPUT VARIABLE	015792
C	WHICH IS THE V(IN) OF	015793
C	THE EXPRESSION --	015794
C		015795
C	V(OUT)	015796
C	----- = IF	015797

C		V(IN)	015796
C			015799
C			015800
C	CRY(3,J) = J*OUT ---	TRANSFER FUNCTION OUTPUT	015801
C		VARIABLE IDENTIFICATION.	015802
C		THIS INTEGER IS A LOCAL	015803
C		IDENTIFICATION INTEGER,	015804
C		REFERENCING (DEPENDING	015805
C		UPON ITYPE) EITHER A	015806
C		SENSOR SIGNAL OR A	015807
C		CONTROLLER OUTPUT VARIABLE	015808
C		WHICH IS THE V(OUT) OF	015809
C		THE EXPRESSION --	015810
C			015811
C		V(OUT)	015812
C		----- = TF	015813
C		V(IN)	015814
C			015815
C	NOTE---IF ABS(ITYPE) EQUALS 8 THEN		015816
C	CRY(2,J) IS GLOBAL REFERENCE INPUT STATE OF Y*		015817
C	(WHOSE FEEDBACK LOOP IS CUT), AND		015818
C	CRY(3,J) IS GLOBAL OUTPUT STATE OF Y*		015819
C			015820
C	CRY(4,J) = KPLUT		015821
C			015822
C		KPLUT = 0 NO PLOTS	015823
C		KPLUT = 1 PLOTS WILL BE MADE	015824
C			015825
C	CRY(5,J) = IAFLE		015826
C			015827
C		THIS INTEGER CONTROL PARAMETER	015828
C		PERMITS THE USER TO SELECT	015829
C		THE CHARACTERISTIC ROOTS FOR A	015830
C		GIVEN TRANSFER FUNCTION FROM	015831
C		EITHER THE CHARACTERISTIC MATRIX,	015832
C		AR, OR ITS TRANSPOSE.	015833
C			015834
C			015835
C			015836
C			015837
C			015838
C			015839
C		IAFLE = 1 PROGRAM WILL USE ROOTS	015840
C		FROM AR TRANSPOSE.	015841
C			015842
C		DEFAULT VALUE IS 0 AND	015843
C		ROOTS FROM AR WILL BE USED.	015844
C			015845
C	NOTE - PROGRAM EXTRACTS ROOTS FOR BOTH		015846
C	AR AND ITS TRANSPOSE. THIS SERVES		015847
C	AS A SORT OF SELF CHECK ON THE		015848
C	ROOT QUALITY. ALTHOUGH IT IS A		015849
C	RARE OCCURRENCE, ROOTS FROM AR		015850
C	TRANSPOSE CAN BE -CLEANER- THAN		015851
C	THOSE OBTAINED FROM AR.		015852
C			015853
C	CRY(6,J) = NO. OF B VARIABLES TO FEED BACK. ITYPE = 7		015854
C		MAX OF 3 B VARIABLES CAN BE FED BACK FOR	015855
C		THE TYPE 7 PSEUDO OPEN LOOP	015856
C		TRANSFER FUNCTION.	015857

C		015858
C	LRY(7,J) = LOCAL ID. OF FIRST B TO RETAIN.	015859
C		015860
C	LRY(8,J) = LOCAL ID. OF SECOND B TO RETAIN.	015861
C		015862
C	LRY(9,J) = LOCAL ID. OF THIRD B TO RETAIN.	015863
C		015864
C		015865
C	-----CALL READIN (LRY, 3, NCYC, 3, KR)	015866
C		015867
C	MATRIX SIZE 3 BY NCYC DEFINING EXPONENT FOR	015868
C	TOLERANCES TOL = (10.)*EXP	015869
C		015870
C	FOR THE JTH CYCLE -	015871
C		015872
C	LRY(1,J) = ROOT TOLERANCE EXPONENT	015873
C	LRY(2,J) = GAIN TOLERANCE EXPONENT	015874
C	LRY(3,J) = ROOT TOLERANCE EXPONENT USED TO	015875
C	REMOVE SHIFT FREQUENCY (SUBROUTINE NUMS)	015876
C		015877
C	NOTE -- IF ROOT OR GAIN LE TOL, THEN	015878
C	SET ROOT OR GAIN EQ 0.0	015879
C		015880
C		015881
C		015882
C		015883
C		015884
C		015885
C		015886
C	GO 500 ICYC=1,NCYC	015887
C		015888
C	IF (ITYPE .EQ. 0) GO TO 500	015889
C		015890
C		015891
C	-----READ(NIT,FORMAT = 20A4) (TITLE(I),I=1,20)	015892
C		015893
C	80 CHARACTER TITLE FOR TRANSFER FUNCTION	015894
C	IDENTIFICATION	015895
C		015896
C		015897
C	-----READ(NIT,FORMAT = 20A4) (LPNAME(I),I=1,5), LPTAPE,LEIGV,LPCLY	015898
C		015899
C	LPNAME(I) PERMITS UP TO 5 FOUR CHARACTER	015900
C	IDENTIFICATIONS WHICH SELECT THE	015901
C	PLOT DISPLAY MODE.	015902
C		015903
C	LPNAME(I) = 4H (ALL BLANKING DISPLAYS	015904
C	ARE IMPLEMENTED --- GO TO 500	015905
C	THE CHARACTERISTIC ROOTS	015906
C	FOR THE SYSTEM ARE FOUND.	015907
C		015908
C	LPNAME(I) = 4H00DE ONLY A BODE DISPLAY.	015909
C		015910
C	= 4HNICH ONLY A NICHOLS DISPLAY.	015911
C		015912
C	= 4HNYQU ONLY A NYQUIST DISPLAY.	015913
C		015914
C	= 4HNINY BOTH NICHOLS AND NYQUIST.	015915
C		015916
C	= 4HBLNN GIVES BODE, NICHOLS, NYQUIST.	015917

C		015918
C	= 4HRCUT GIVES A ROOT LOCUS DISPLAY.	015919
C		015920
C	LPTAPE = 4H7TRK DUMP /PSTUFF/ ON 7-TRACK TAPE	015921
C	UNIT 12 IN JCL. DON'T IF.NE.4H7TRK	015922
C		015923
C	LEIGV = 4HLEIGV COMPUTE EIGENVECTORS	015924
C	LPOLY = 4HPOLY COMPUTE AND PRINT TRANSFER	015925
C	FUNCTION AS RATIO OF POLYNOMIALS	015926
C		015927
C	IF (LEIGV .NE. 4HLEIGV) GO TO 23	015928
C		015929
C	-----READ(NIT,FORMAT = 2F10.0) FCMIN, FCMAX	015930
C		015931
C	23 CONTINUE	015932
C		015933
C	COMPUTE EIGENVECTORS ASSOCIATED WITH EIGENVALUES HAVING	015934
C	IMAGINARY PART .GT. FCMIN AND .LT. FCMAX	015935
C		015936
C	EIGENVECTORS ASSOCIATED WITH REPEATED EIGENVALUES	015937
C	CANNOT BE FOUND WITH PRESENT SUBROUTINE LIGVEC	015938
C		015939
C		015940
C		015941
C		015942
C		015943
C	GO 500 TOP=1.5	015944
C		015945
C		015946
C	IF (LPNAME(IOP) .EQ. 4H) GO TO 500	015947
C		015948
C	IF (LPNAME(IOP) .EQ. 4HBBDE	015949
C	*.OR. LPNAME(IOP) .EQ. 4HNICH	015950
C	*.OR. LPNAME(IOP) .EQ. 4HNYGU	015951
C	*.OR. LPNAME(IOP) .EQ. 4HNINY	015952
C	*.OR. LPNAME(IOP) .EQ. 4HBCAN) GO TO 200	015953
C		015954
C	IF (LPNAME(IOP) .EQ. 4HRCCT) GO TO 500	015955
C		015956
C		015957
C	200 CONTINUE	015958
C		015959
C		015960
C	*** FREQUENCY RESPONSE SECTION ***	015961
C		015962
C		015963
C	-----READ(NIT,FORMAT = 6F10.0) FMIN, FMAX, UDMIN, UDMAX, AMIN, AMAX	015964
C		015965
C	FMIN = FREQUENCY SWEEP LOWER LIMIT	015966
C	FMAX = FREQUENCY SWEEP UPPER LIMIT	015967
C	UDMIN = MINIMUM DB AMPLITUDE FOR BODE, NICHOLS PLOTS	015968
C	UDMAX = MAXIMUM DB AMPLITUDE FOR BODE, NICHOLS PLOTS	015969
C	AMIN = MINIMUM AMPLITUDE FOR NYQUIST PLOTS	015970
C	AMAX = MAXIMUM AMPLITUDE FOR NYQUIST PLOTS	015971
C		015972
C		015973
C	GO TO 500	015974
C		015975
C		015976
C	300 CONTINUE	015977

C		015978	
C		015979	
C		015980	
C		015981	
C		015982	
C		015983	
C		015984	
C		015985	
C	*** ROOT LOCUS SECTION ***	015986	
C		015987	
C		015988	
C	-----CALL READIM (IJM, 2, NRLC, 2, KR)	015989	
C		015990	
C	MATRIX SIZE 2 BY NRLC FOR ROOT LOCUS PLOT CONTROL	015991	
C		015992	
C	NRLC = NUMBER OF ROOT LOCI TO PERFORM	015993	
C		015994	
C	FOR THE JTH ROOT LOCI -	015995	
C		015996	
C	IJM(1,J) = ISNIM = 1 STARTING POINT IS	015997	
C		OPEN LOOP ZERO.	015998
C		= 2 STARTING POINT IS	015999
C		OPEN LOOP POLE.	016000
C		= 3 STARTING POINT IS	016001
C		CLOSED LOOP POLE.	016002
C			016003
C			016004
C	IJM(2,J) = ELEMENT LOCATION IN ROOT ARRAY	016005	
C	FOR STARTING ROOT LOCI.	016006	
C		016007	
C		016008	
C	-----CALL READ (W1, 6, NRLC, KR, KP)	016009	
C		016010	
C	MATRIX SIZE 6 BY NRLC FOR ROOT LOCUS CONTROL DATA	016011	
C		016012	
C	FOR THE JTH ROOT LOCI -	016013	
C		016014	
C	W1(1,J) = THETAD(J) INITIAL SEARCH ANGLE,	016015	
C	NORMALLY -180. (DEGREES)	016016	
C		016017	
C	W1(2,J) = SCL SCALE FACTOR, NORMALLY	016018	
C	SCL = 1.0	016019	
C		016020	
C	W1(3,J) = ALLOC PHASE CONTROL PARAMETER.	016021	
C	ALLOC = +1. --- 180. DEG. PHASE.	016022	
C	ALLOC = -1. --- 0 DEG. PHASE.	016023	
C		016024	
C	W1(4,J) = XMIN MIN REAL VALUE TO PLOT.	016025	
C		016026	
C	W1(5,J) = XMAX MAX REAL VALUE TO PLOT.	016027	
C		016028	
C	W1(6,J) = YMAX MAX IMAG VALUE TO PLOT AND	016029	
C	YMIN SET TO - YMAX.	016030	
C		016031	
C		016032	
C	500 CONTINUE	016033	
C		016034	
C	RETURN	016035	
C	RETURN	016036	
C	RETURN	016037	

C	RETURN	016038
C	RETURN	016039
C	RETURN	016040
C	RETURN	016041
	RETURN	016042
	END	016043

	SUBROUTINE DEF4	016044
C		016045
C	DEBUG #115	016046
C		016047
C		016048
C		016049
C	*****	016050
C	*****	016051
C	*	016052
C	DESCRIPTION OF PROGRAM DISCOS OUTPUT DATA	016053
C	*	016054
C	*****	016055
C	*****	016056
C		016057
C		016058
C	----- COMMENTS (FORMAT 13AG)	016059
C		016060
C	COMMENT CARDS READ INTO THE	016061
C	PROGRAM WITH SUBROUTINE COMMENT	016062
C	ARE PRINTED HERE.	016063
C		016064
C		016065
C		016066
C	*****	016067
C	*	016068
C	SUBROUTINE DYNSTD OUTPUT	016069
C	*	016070
C	*****	016071
C		016072
C		016073
C		016074
C		016075
C	----- SUMMARY OF DYNAMIC-SIMULATION INPUT DATA * * * * *	016076
C		016077
C		016078
C	----- ACTUAL MAXIMUM INTEGRATION GRAVITY GRADIENT	016079
C	----- SIZES SIZES DATA DATA	016080
C	-----	016081
C		016082
C	----- NB = NBMAX = STARTT = G1 = GAM1 =	016083
C		016084
C	----- NH = NHMAX = DLTAT = G2 = GAM2 =	016085
C		016086
C	----- NSPT = NSPMAX = ENDTAT = G3 = GAM3 =	016087
C		016088
C	----- NUFMO = NMWMAX = GMAJ = RMAG =	016089
C		016090
C	----- NULLTA = NMWBDU =	016091

C				016092
C	----- NU	= NM0000 =	MISC. DATA	016093
C			-----	016094
C	----- NBETA	= KMU =		016095
C				016096
C	----- NLAM	= KY =	NOBINT =	016097
C				016098
C	----- NEQ	= KU =	NOPLUT =	016099
C				016100
C	-----		IFLNER =	016101
C				016102
C				016103
C				016104
C				016105
C				016106
C				016107
C				016108
C				016109
C			THE FOLLOWING LIST IDENTIFIES THE	016110
C			OUTPUT VARIABLES SUMMARIZED ON THE	016111
C			PREVIOUS PAGE.	016112
C				016113
C	*****			016114
C				016115
C				016116
C		NU	= NO. OF BODIES.	016117
C		NH	= NO. OF HINGES.	016118
C				016119
C		NSPT	= NO. OF SENSOR POINTS.	016120
C				016121
C		NDELTA	= NO. OF CONTROL SYSTEM DELTAS.	016122
C				016123
C		NU	= NO. OF U'S.	016124
C				016125
C		NBETA	= NO. OF BETA'S.	016126
C				016127
C		NLAM	= NO. OF LAMBDA'S (CONSTRAINTS).	016128
C				016129
C		NEQ	= NO. OF STATE EQUATIONS.	016130
C				016131
C		NBMAX	= MAXIMUM DIMENSIONED NO. OF BODIES.	016132
C				016133
C		NHMAX	= MAXIMUM DIMENSIONED NO. OF HINGES.	016134
C				016135
C		NSPMAX	= MAXIMUM DIMENSIONED NO. OF SENSOR POINTS.	016136
C				016137
C		NMWMAX	= MAXIMUM DIMENSIONED NO. OF MOM. WHEELS.	016138
C				016139
C		NMW000	= MAXIMUM DIMENSIONED NO. OF MOM. WHEELS	016140
C			PER BODY.	016141
C				016142
C		NM0000	= MAXIMUM DIMENSIONED NO. OF MOVES PER BODY.	016143
C				016144
C		KMU	= MAXIMUM DIMENSIONED NO. OF U'S PER BODY,	016145
C			= 0 + NM0000 + NMW000.	016146
C				016147
C				016148
C		KY	= MAXIMUM DIMENSIONED SIZE FOR STATE VECTOR.	016149
C				016150
C		KU	= MAXIMUM DIMENSIONED NO. OF U'S,	016151

C	= N0MAX*(0 + NMDJ0D) + N4#MAX.	016152
C		016153
C		016154
C		016155
C		016156
C		016157
C		016158
C	OUTPUT VARIABLE IDENTIFICATION SUMMARY	016159
C	(CONT'D)	016160
C		016161
C		016162
C	*****	016163
C		016164
C		016165
C		016166
C	STARTT = START TIME FOR TIME RESPONSE.	016167
C		016168
C	DELTAT = INTEGRATION STEP SIZE.	016169
C		016170
C	ENDT = END TIME FOR TIME RESPONSE.	016171
C		016172
C		016173
C		016174
C		016175
C		016176
C		016177
C		016178
C		016179
C	G1 = X COMPONENT OF GRAVITY VECTOR. (INPUT)	016180
C		016181
C	G2 = Y COMPONENT OF GRAVITY VECTOR. (INPUT)	016182
C		016183
C	G3 = Z COMPONENT OF GRAVITY VECTOR. (INPUT)	016184
C		016185
C	GMAG = SQRT(G1**2 + G2**2 + G3**2) --	016186
C	ACCELERATION OF GRAVITY	016187
C		016188
C	GAM1 = DIRECTION COSINE (GRAVITY VECTOR AND X).	016189
C		016190
C	GAM2 = DIRECTION COSINE (GRAVITY VECTOR AND Y).	016191
C		016192
C	GAM3 = DIRECTION COSINE (GRAVITY VECTOR AND Z).	016193
C		016194
C	ROMAG = REFERENCE RADIUS FOR ACTING GRAVITY VECTOR.	016195
C	(INPUT)	016196
C		016197
C	NOPRINT = MULTIPLE OF DELTAT TO PRINT TIME RESPONSE.	016198
C		016199
C	NOPLCT = MULTIPLE OF DELTAT TO WRITE PLOT TAPE.	016200
C		016201
C	IFELNR = 0 (NON LINEAR TIME RESPONSE)	016202
C	= 1 (LINEAR ANALYSIS OR FREQ RESPONSE)	016203
C		016204
C		016205
C		016206
C		016207
C		016208
C		016209
C	THE FOLLOWING SERIES OF ARRAYS PERTAIN	016210
C	TO THE MODEL SIMULATION AND ARE PRINTED	016211

```

C                                OUT AT THE BEGINNING OF EACH SIMULATION                                016212
C                                                                    016213
C*****                                                                    016214
C                                                                    016215
C                                                                    016216
C-----THE TOPOLOGY ARRAY (ITOPOL) FOR THIS CASE FOLLOWS                                016217
C                                                                    016218
C              (1)   (2)   ...   (NH)                                                                    016219
C                                                                    016220
C-----      1     1                                                                    016221
C-----      2     1                                                                    016222
C                                                                    016223
C                                THIS IS THE INPUT INTEGER ARRAY ITOPOL                                016224
C                                (SEE PREVIOUS CHAPTER -- INPUT DATA)                                016225
C                                                                    016226
C                                                                    016227
C-----THE CONSTRAINT SPECIFICATIONS FOR THIS CASE FOLLOW                                016228
C                                                                    016229
C              (1)   (2)   ...   (NH)                                                                    016230
C                                                                    016231
C-----      1     1                                                                    016232
C-----      2     1                                                                    016233
C-----      3     1                                THIS IS THE INPUT INTEGER ARRAY IHDATA                                016234
C-----      4     1                                (SEE PREVIOUS CHAPTER -- INPUT DATA)                                016235
C-----      5     1                                                                    016236
C-----      6     1                                                                    016237
C-----      7     1                                                                    016238
C                                                                    016239
C                                                                    016240
C-----THE SPECIFIED INITIAL HINGE ANGLES                                016241
C      AND DISPLACEMENTS (BETAH) FOLLOW                                016242
C                                                                    016243
C              (1)   (2)   ...   (NH)                                                                    016244
C                                                                    016245
C-----      1     1                                                                    016246
C-----      2     1                                ROWS 1-3 = HINGE ANGLES (CONSISTENT WITH                                016247
C                                                                    016248
C-----      .     .                                ITYPE).                                016249
C-----      .     .                                ROWS 4-6 = HINGE DISPLACEMENTS- Q RELATIVE TO P                                016250
C-----      .     .                                                                    016251
C-----      6     1                                                                    016252
C                                                                    016253
C                                                                    016254
C                                                                    016255
C                                                                    016256
C                                                                    016257
C                                                                    016258
C                                                                    016259
C-----THE SPECIFIED INITIAL HINGE RATES (BETAHD) FOLLOW                                016260
C                                                                    016261
C              (1)   (2)   ...   (NH)                                                                    016262
C                                                                    016263
C-----      1     1                                                                    016264
C-----      2     1                                ROWS 1-3 = ANGULAR RATES.                                016265
C-----      .     .                                                                    016266
C-----      .     .                                ROWS 4-6 = DISPLACEMENT RATES- Q RELATIVE TO P.                                016267
C-----      .     .                                                                    016268
C-----      6     1                                                                    016269
C                                                                    016270
C                                                                    016271

```

C-----THE NO. OF ELASTIC MODES/BODY ARRAY (IROPX) FOLLOWS	016272
C	016273
C (1) (2) ... (NB)	016274
C	016275
C-----1 1 THE JTH ENTRY IS THE NO. OF ELASTIC	016276
C MODES FOR BODY J.	016277
C	016278
C	016279
C-----THE NO. OF P/Q HINGE POINTS/BODY ARRAY (NHPOI) FOLLOWS	016280
C	016281
C (1) (2) ... (NB)	016282
C	016283
C-----1 1 ELEMENTS ARE THE NO. OF P/Q	016284
C HINGE POINTS ON EACH BODY.	016285
C	016286
C	016287
C-----THE NO. OF SENSOR POINTS/BODY ARRAY (NSPOI) FOLLOWS	016288
C	016289
C (1) (2) ... (NB)	016290
C	016291
C-----1 1 ELEMENTS ARE THE NO. OF SENSOR	016292
C POINTS ON EACH BODY.	016293
C	016294
C	016295
C-----THE MOM. WHEEL/BODY TABLE (NMOW) FOLLOWS	016296
C	016297
C (1) (2) ... (NB)	016298
C	016299
C-----COL J = JTH BODY	016300
C-----1 1 ROW 1 = NO. OF MOM. WHEELS ON BODY J.	016301
C-----2 1 ROW 2 = NO. OF VARIABLE SPEED WHEELS ON BODY J.	016302
C----- . . ROW 3 = SUCCESSIVE ROWS ARE THE MOMENTUM WHEEL	016303
C NUMBERS	016304
C----- . . ON BODY J (IN ASCENDING ORDER).	016305
C-----2+NMWBOO 1 .	016306
C	016307
C	016308
C	016309
C	016310
C	016311
C	016312
C	016313
C	016314
C-----THE STATE VECTOR LENGTH ARRAY (LENU) FOLLOWS	016315
C	016316
C (1) (2) ... (2*NB + 2)	016317
C	016318
C-----1 1	016319
C	016320
C THE ELEMENTS ARE THE LENGTHS OF SEGMENTS	016321
C OF THE STATE VECTOR.	016322
C	016323
C ORDER IS	016324
C	016325
C U(1),U(2),...,U(NB),X1(1),...,X1(NB),BETA,DELTA	016326
C	016327
C	016328
C-----THE STATE VECTOR LOCATION ARRAY (LOCU) FOLLOWS	016329
C	016330
C (1) (2) ... (2*NB + 2)	016331

C				016332
C-----	1	1		016333
C				016334
C			LEADING ELEMENT LOCATION IN THE STATE VECTOR	016335
C			FOR THE SEGMENTS DESCRIBED IN ARRAY (LENU).	016336
C				016337
C				016338
C-----	THE SPECIFIED SENSOR POINT/BODY CORRELATION ARRAY (IFTSMW)			016339
C			FOLLOWS	016340
C				016341
C			(1) (2) ... (NSPT)	016342
C				016343
C-----	1	1	THE JTH ELEMENT IS THE BODY NO. ON	016344
C			WHICH SENSOR POINT J IS LOCATED.	016345
C				016346
C				016347
C				016348
C				016349
C				016350
C				016351
C				016352
C-----	THE FOLLOWING DATA IS SPECIFIED MOM. WHEEL INFORMATION			016353
C			(IF ANY)	016354
C			AND CONTROLLER INFORMATION	016355
C				016356
C				016357
C-----	THE SPECIFIED MOM. WHEEL CONTROL ARRAY (IMU) FOLLOWS			016358
C				016359
C			(1) (2) ... (NCFMO)	016360
C				016361
C			COL J = JTH MOMENTUM WHEEL	016362
C-----	1	1	ROW 1 = WHEEL SENSOR POINT NO.	016363
C-----	2	1	ROW 2 = SPIN AXIS	016364
C-----	3	1	ROW 3 = 1 ACTIVE	016365
C			= 0 CONSTANT SPEED	016366
C				016367
C				016368
C-----	THE SPECIFIED MOM. WHEEL RATES AND INERTIAS (AMU) FOLLOW			016369
C				016370
C			(1) (2) ... (NCFMO)	016371
C				016372
C			COL J = JTH MOMENTUM WHEEL	016373
C-----	1	1	ROW 1 = INITIAL WHEEL SPIN RATE	016374
C-----	2	1	ROW 2 = SPIN INERTIA	016375
C				016376
C				016377
C-----	THE SPECIFIED CONTROLLER INITIAL CONDITIONS			016378
C			AND CHARACTERISTICS FOLLOW	016379
C			(THE FIRST NDELTA ARE INITIAL CONTROLLER STATE	016380
C			VARIABLES, THERE ARE K ADDITIONAL PARAMETERS)	016381
C				016382
C			THIS IS THE USER INPUT ARRAY CNTDTA.	016383
C				016384
C			THE ADDITIONAL K PARAMETERS, IF ANY, ARE	016385
C			AVAILABLE TO THE USER FOR USER-PAK DATA.	016386
C				016387
C			THE FIRST NDELTA ENTRIES IN THIS ARRAY	016388
C			ARE THE INITIAL CONDITIONS FOR THE	016389
C			CONTROL VARIABLES.	016390
C				016391

```

C
C
C*****
C*****
C
C
C
C
C
C
C*****
C*
C*   SUBROUTINE MRIGID -- OUTPUT FOR RIGID BODY
C*
C*****
C
C           THE FOLLOWING IS TYPICAL FOR THE ITH BODY
C
C----- SUMMARY OF INPUT DATA FOR BODY 1 WHICH IS RIGID.
C
C-----THE 6X6 INERTIA MATRIX IS --
C
C           (1)      (2)      (3)      (4)      (5)      (6)
C-----
C----- 1  1  IXX  -IXY  -IXZ   0  -SZ   SY
C----- 2  1 -IYX  IYY  -IYZ   SZ   0  -SX
C----- 3  1 -IZX  -IZY  IZZ  -SY   SX   0
C----- 4  1   0   SZ  -SY   M   0   0
C----- 5  1 -SZ   0   SX   0   M   0
C----- 6  1   SY  -SX   0   0   0   M
C
C
C-----FOR BODY 1 THE P-Q HINGE NO. AND THE EULER ROTATION TYPE
C   APPEAR IN THE FOLLOWING INTEGER ARRAY WHICH IS FOLLOWED BY
C   AN ARRAY CONTAINING EULER ANGLES (1,2,3), AND POSITION
C   VECTOR COMPONENTS (4,5,6) THAT POSITION THE HINGE TRIAD
C   WRT THE BODY TRIAD
C
C           ----- DATA HERE -----
C
C   IF BODY 1 HAS ANY SENSOR POINTS,
C   THE FOLLOWING WILL BE PRINTED
C
C-----FOR BODY 1 THE SENSOR POINT NO. AND THE EULER ROTATION TYPE
C   APPEAR IN THE FOLLOWING INTEGER ARRAY WHICH IS FOLLOWED
C   BY AN ARRAY CONTAINING EULER ANGLES (1,2,3), AND POSITION
C   VECTOR COMPONENTS (4,5,6) THAT POSITION THE SENSOR TRIAD
C   WRT THE BODY TRIAD
C
C           ----- DATA HERE -----
C
C*****
C*****
C
C
C
C
C
C

```

```

016392
016393
016394
016395
016396
016397
016398
016399
016400
016401
016402
016403
016404
016405
016406
016407
016408
016409
016410
016411
016412
016413
016414
016415
016416
016417
016418
016419
016420
016421
016422
016423
016424
016425
016426
016427
016428
016429
016430
016431
016432
016433
016434
016435
016436
016437
016438
016439
016440
016441
016442
016443
016444
016445
016446
016447
016448
016449
016450
016451

```

C	016452
C*****	016453
C*	016454
C SUBROUTINE MSMODC - OUTPUT FOR FLEXIBLE BODY,	016455
C CONSISTENT MASS MATRIX	016456
C*	016457
C*****	016458
C	016459
C	016460
C	016461
C	016462
C	016463
C THIS SUBROUTINE PRINTS OUT SEVERAL MATRICES THAT ARE	016464
C RELATED TO THE FORM OF THE GOVERNING EQUATIONS.	016465
C	016466
C THE READER WILL BE REFERRED TO VOL I AND THE INPUT	016467
C DATA STREAM FOR FURTHER CLARIFICATION.	016468
C	016469
C-----SUMMARY OF INPUT DATA FOR BODY I WHICH IS	016470
C FLEXIBLE W/CONSISTENT MASS MATRIX.	016471
C	016472
C	016473
C-----THE INPUT PARAMETERS--- IFRBM, IFDIK, IFDIAD ARE	016474
C	016475
C SEE INPUT DATA FOR MSMODC	016476
C	016477
C	016478
C-----THE JOUF TABLE FOLLOWS---	016479
C	016480
C DEGREE OF FREEDOM AS INPUT	016481
C	016482
C SEE INPUT DATA FOR MSMODC	016483
C	016484
C	016485
C-----THE MODE SELECTION VECTOR FOLLOWS	016486
C	016487
C MODE SELECTION VECTOR AS INPUT	016488
C	016489
C SEE INPUT DATA FOR MSMODC	016490
C	016491
C	016492
C-----FOR BODY NO. 1 THE POSITION VECTOR FROM THE BODY ORIGIN	016493
C TO JOINT K IS	016494
C	016495
C X = Y = Z =	016496
C	016497
C WHERE K IS JOINT COORDINATES	016498
C USED TO DEVELOP RIGID BODY MODES	016499
C	016500
C SEE INPUT DATA FOR MSMODC	016501
C	016502
C	016503
C	016504
C	016505
C	016506
C	016507
C-----THE CONSISTENT, REPARTITIONED MASS MATRIX IS--	016508
C	016509
C THIS IS THE REPARTITIONED MASS	016510
C MATRIX AND IS CONSISTENT WITH	016511

C	THE JOGF TABLE.	016512
C		016513
C		016514
C	-----THE REPARTITIONED MODAL MATRIX IS---	016515
C		016516
C	THIS IS THE REPARTITIONED MODAL	016517
C	MATRIX. THE ROWS ARE CONSISTENT	016518
C	WITH THE REPARTITIONED MASS MATRIX	016519
C	AND THE COLS ARE CONSISTENT WITH	016520
C	THE ELEMENTS OF THE MODE SELECTION	016521
C	VECTOR.	016522
C		016523
C		016524
C	-----THE UNDEFORMED- INERTIA MATRIX (MO) IS---	016525
C		016526
C	THIS IS THE MO MATRIX NOTED AS	016527
C	EQ. 11-87 OF VOL I	016528
C		016529
C		016530
C	-----	016531
C	THERE THEN FOLLOWS MATRICES	016532
C		016533
C	A COEFFICIENTS	016534
C	B COEFFICIENTS	016535
C	COFX1 COEFFICIENTS	016536
C	COFX2 COEFFICIENTS	016537
C	COFY1 COEFFICIENTS	016538
C		016539
C	WHICH ARE THE ALPHA, B, AND C COEFFICIENTS	016540
C	GIVEN AS EQ. 11-88 OF VOL I	016541
C		016542
C		016543
C	-----	016544
C	THERE THEN FOLLOWS MATRICES	016545
C		016546
C	C11, C22, C33,	016547
C	C12, C13, C23	016548
C		016549
C	WHICH ARE IDENTIFIED AS EQ. 11-89 OF VOL I	016550
C		016551
C		016552
C		016553
C		016554
C		016555
C		016556
C		016557
C	-----THE MODAL STIFFNESS IS --	016558
C	----- DATA HERE -----	016559
C		016560
C		016561
C		016562
C	-----THE MODAL DAMPING MATRIX IS --	016563
C	----- DATA HERE -----	016564
C		016565
C		016566
C		016567
C		016568
C	THERE FOLLOWS TWO ARRAYS CONTAINING THE	016569
C		016570
C	INITIAL MODAL DISPLACEMENTS AND	016571

C	INITIAL MODAL VELOCITIES RESPECTIVELY.	016572
C		016573
C		016574
C	-----FOR BODY I THE P-Q HINGE NO., THE EULER ROTATION TYPE	016575
C	AND THE JOINT NO. CORRESPONDING TO THE P-Q HINGE	016576
C	APPEAR IN THE FOLLOWING INTEGER ARRAY WHICH IS FOLLOWED BY AN	016577
C	ARRAY CONTAINING EULER ANGLES THAT POSITION THE HINGE	016578
C	TRIAD WRT THE BODY TRIAD	016579
C		016580
C	---- DATA HERE ----	016581
C		016582
C		016583
C	IF BODY I HAS ANY SENSOR POINTS,	016584
C	THE FOLLOWING WILL BE PRINTED	016585
C		016586
C		016587
C	-----FOR BODY I THE SENSOR POINT NO., THE EULER ROTATION TYPE	016588
C	AND THE JOINT NO. CORRESPONDING TO THE SENSOR POINT	016589
C	APPEAR IN THE FOLLOWING INTEGER ARRAY WHICH IS FOLLOWED	016590
C	BY AN ARRAY CONTAINING EULER ANGLES THAT POSITION THE	016591
C	SENSOR TRIAD WRT THE BODY TRIAD	016592
C		016593
C	---- DATA HERE ----	016594
C		016595
C		016596
C		016597
C	*****	016598
C	*****	016599
C		016600
C		016601
C		016602
C		016603
C		016604
C	*****	016605
C	*	016606
C	SUBROUTINE MSMODEL - OUTPUT FOR FLEXIBLE BODY, LUMPED MASS MATRIX	016607
C	*	016608
C	*****	016609
C		016610
C		016611
C	TYPICAL OUTPUT FOR THE ITH BODY	016612
C		016613
C		016614
C	THIS SUBROUTINE PRINTS OUT SEVERAL MATRICES THAT ARE	016615
C	RELATED TO THE FORM OF THE GOVERNING EQUATIONS.	016616
C		016617
C		016618
C	THE READER WILL BE REFERRED TO VOL I AND THE INPUT	016619
C	DATA STREAM FOR FURTHER CLARIFICATION.	016620
C		016621
C		016622
C	-----OUTPUT MATRICES INER0, STAT0, MASS0, JOCE0F, AUCOE0F, ECOE0F	016623
C		016624
C	ARE THE J, -S, M, D, A, E, PARTITIONS	016625
C	RESPECTIVELY OF THE MATRIX M0 OF EQ. II-87	016626
C	OF VOL I	016627
C		016628
C		016629
C	-----OUTPUT MATRIX M00 IS MATRIX M0 OF EQ. II-87 OF VOL I	016630
C		016631

C		016632
C	-----OUTPUT MATRICES ACOF, BCOF, CXY, CXZ, CYZ,	016633
C		016634
C	ARE THE ALPHA, B, AND C	016635
C	COEFFICIENTS IN THE MATRIX	016636
C	GIVEN AS EQ. 11-88 OF VOL I	016637
C		016638
C		016639
C		016640
C		016641
C		016642
C		016643
C		016644
C		016645
C		016646
C	-----OUTPUT MATRICES C11, C22, C33,	016647
C	C12, C13, C23	016648
C		016649
C	ARE THE CONSTITUENTS OF THE	016650
C	MATRIX GIVEN AS EQ. 11-89 OF VOL I	016651
C		016652
C		016653
C	-----OUTPUT MATRIX XEO	016654
C		016655
C	CONTAINS THE INITIAL MODAL	016656
C	DEFLECTIONS. (AS INPUT)	016657
C		016658
C		016659
C	-----OUTPUT MATRIX XEOV	016660
C		016661
C	CONTAINS THE INITIAL MODAL	016662
C	VELOCITIES. (AS INPUT)	016663
C		016664
C		016665
C	-----FOR BODY I THE P-Q HINGE NO., THE EULER ROTATION TYPE	016666
C	AND THE JOINT NO. CORRESPONDING TO THE P-Q HINGE	016667
C	APPEAR IN THE FOLLOWING INTEGER ARRAY WHICH IS FOLLOWED BY AN	016668
C	ARRAY CONTAINING EULER ANGLES THAT POSITION THE HINGE	016669
C	TRIAD WRT THE BODY TRIAD	016670
C		016671
C	----- DATA HERE -----	016672
C		016673
C		016674
C	IF BODY I HAS ANY SENSOR POINTS,	016675
C	THE FOLLOWING WILL BE PRINTED	016676
C		016677
C		016678
C	-----FOR BODY I THE SENSOR POINT NO., THE EULER ROTATION TYPE	016679
C	AND THE JOINT NO. CORRESPONDING TO THE SENSOR POINT	016680
C	APPEAR IN THE FOLLOWING INTEGER ARRAY WHICH IS FOLLOWED	016681
C	BY AN ARRAY CONTAINING EULER ANGLES THAT POSITION THE	016682
C	SENSOR TRIAD WRT THE BODY TRIAD	016683
C		016684
C	----- DATA HERE -----	016685
C		016686
C		016687
C		016688
C	*****	016689
C	*****	016690
C		016691

C		016692
C		016693
C		016694
C		016695
C	IF THE USER UTILIZES POLYNOMIAL INPUT FOR	016696
C	CONTROL SYSTEM TRANSFER FUNCTIONS, THE FOLLOWING	016697
C	MATRIX WILL BE PRINTED.	016698
C		016699
C		016700
C	-----OUTPUT MATRIX CPLY(KPLY,2*NPLY)	016701
C		016702
C	WHERE KPLY = ROW DIMENSION SIZE OF CPLY IN	016703
C	SUBROUTINE CONTRL	016704
C		016705
C	NPLY = NO. OF INPUT POLYNOMIAL RATIOS	016706
C		016707
C	AND FOR I = ODD INTEGER --	016708
C		016709
C	COL I = DENOMINATOR POLYNOMIAL COEFFICIENTS	016710
C	IN ASCENDING ORDER	016711
C		016712
C	COL I+1 = NUMERATOR POLYNOMIAL COEFFICIENTS	016713
C	IN ASCENDING ORDER	016714
C		016715
C		016716
C	-----THE FOLLOWING INTEGER ARRAY (INDEP) PRESCRIBES INDEPENDENT	016717
C	VARIABLES (I) AND DEPENDENT VARIABLES (O)	016718
C		016719
C	THE ELEMENTS OF THIS ARRAY IDENTIFY	016720
C	WHICH VARIABLES SURVIVE IN THE FINDU	016721
C	SEARCH TO DETERMINE AN INDEPENDENT	016722
C	SET TO BE INTEGRATED.	016723
C		016724
C	THE SURVIVING VARIABLES WILL ALSO	016725
C	REPRESENT THE FIRST NX ROWS OF	016726
C	THE LINEARIZED MATRIX, A, USED	016727
C	BY SUBROUTINE DYNSSDD.	016728
C		016729
C	NOTE - NX = NO. OF NON-ZERO ENTRIES	016730
C	IN ARRAY INDEP	016731
C		016732
C		016733
C		016734
C		016735
C		016736
C	*****	016737
C	*****	016738
C		016739
C		016740
C		016741
C		016742
C		016743
C	*****	016744
C	C*	016745
C	C* SUBROUTINE DYNSSDD OUTPUT	016746
C	C*	016747
C	*****	016748
C		016749
C		016750
C	THE PRINTOUT IS TYPICAL FOR A GIVEN SIMULATION TIME, T.	016751

C		016752
C	THE T=0 PRINT OUT IS ALWAYS GIVEN	016753
C	(EVEN FOR A LINEARIZED ANALYSIS)	016754
C		016755
C		016756
C	THE DATA ARE PRESENTED IN VECTOR FORM AND	016757
C	ORDERED AS FOLLOWS---	016758
C		016759
C	-----THE STATE VECTOR Y =	016760
C		016761
C	-----THE STATE VECTOR TIME DERIVATIVE YDT =	016762
C		016763
C	-----THE DELTAS (EULER ANGLES, POSITION COORDINATES) ARE	016764
C		016765
C	-----THE DELTA TIME DERIVATIVES ARE	016766
C		016767
C	-----THE DELTAS (CONTROL SYSTEM VARIABLES) ARE	016768
C		016769
C	-----THE DELTA TIME DERIVATIVES ARE	016770
C		016771
C		016772
C		016773
C		016774
C		016775
C		016776
C		016777
C		016778
C		016779
C	THE FOLLOWING INFORMATION IS TYPICAL FOR BODY 1.	016780
C		016781
C		016782
C	-----FOR BODY 1 THE VELOCITIES ARE	016783
C		016784
C	ORDER IS DMGX	016785
C	DMSY BODY ANGULAR VELOCITY	016786
C	DMGZ	016787
C		016788
C	U	016789
C	V BODY TRANSLATIONAL VELOCITY	016790
C	W	016791
C		016792
C	XID(1)	016793
C	XID(2)	016794
C	. BODY MODAL VELOCITIES	016795
C	.	016796
C	.	016797
C	XID(NEL)	016798
C		016799
C	THETA(1)	016800
C	. MOMENTUM WHEEL ANGULAR VELOCITY	016801
C	. (RELATIVE TO SENSOR POINT TRIAD)	016802
C	. (ACTIVE WHEEL)	016803
C		016804
C		016805
C	-----FOR BODY 1 THE CORRESPONDING MOMENTA ARE	016806
C		016807
C	ORDER IS HX	016808
C	HY BODY AXES REF. ANGULAR MOMENTUM	016809
C	HZ INCLS CONTRIBUTION OF CONST. SPEED WHEEL	016810
C		016811

C	LX	016812
C	LY BODY AXES REF. LINEAR MOMENTUM	016813
C	LZ	016814
C		016815
C	P XI(1)	016816
C	P XI(2)	016817
C	. BODY AXES REF. MODAL MOMENTUM	016818
C	.	016819
C	.	016820
C	P XI(NE)	016821
C		016822
C	H MW(1)	016823
C	. BODY AXES REF. MOM WHEEL MOMENTUM	016824
C	. (ACTIVE WHEEL)	016825
C	.	016826
C		016827
C		016828
C		016829
C		016830
C		016831
C		016832
C		016833
C		016834
C	-----FOR BODY I ITS CONTRIBUTION TO TOTAL ANGULAR	016835
C	AND LINEAR MOMENTUM IS	016836
C		016837
C	ORDER IS HX	016838
C	HY ANGULAR MOMENTUM	016839
C	HZ	016840
C	(REFERENCED TO INERTIAL ORIGIN)	016841
C	LX	016842
C	LY LINEAR MOMENTUM	016843
C	LZ	016844
C		016845
C		016846
C	-----FOR BODY I ITS CONTRIBUTION TO TOTAL KINETIC AND	016847
C	POTENTIAL ENERGIES IS	016848
C		016849
C		016850
C	-----FOR BODY I THE ELASTIC DEFLECTIONS ARE	016851
C		016852
C		016853
C	-----FOR BODY I TRANSFORMATION MATRIX TO BODY I COORDINATES IS	016854
C		016855
C	USED TO TRANSFORM VECTORS AND TENSORS	016856
C	FROM BODY I FIXED REFERENCE	016857
C	TO BODY I FIXED REFERENCE	016858
C		016859
C		016860
C	-----FOR BODY I CM LOCATION WRT BODY I REFERENCE POINT AND	016861
C	COORDINATES IS	016862
C		016863
C		016864
C	-----FOR BODY I INERTIA TENSOR RELATIVE TO ITS CENTER OF MASS	016865
C	WRT BODY I COORDINATES IS	016866
C		016867
C		016868
C	-----THE INTERCONNECTION CONSTRAINT FORCES (LAMBDAS) ARE	016869
C		016870
C		016871

B-295

C*****	016932
C*****	016933
C	016934
C	016935
C	016936
C	016937
C	016938
C	016939
C*****	016940
C*	016941
C* SUBROUTINE DYN540 OUTPUT - LINEARIZED SYSTEM ANALYSIS	016942
C*	016943
C*****	016944
C	016945
C	016946
C-----OUTPUT MATRIX -A- (NJQ,NX)	016947
C	016948
C	016949
C NJQ = TOTAL NO. OF EQUATIONS	016950
C LINEARIZED (INCLUDING	016951
C AUXILIARY EQUATIONS FROM	016952
C SUBROUTINE EQADD)	016953
C	016954
C NX = NO OF INDEPENDENT STATE EQUATIONS	016955
C DETERMINED BY FINOJ IN DYN520.	016956
C	016957
C	016958
C THIS IS THE MATRIX OF PARTIAL DERIVATIVES	016959
C WHICH ARE THE LINEARIZED COMPONENTS OF THE	016960
C STATE VARIABLES AS DETERMINED BY	016961
C SUBROUTINE LINEAR.	016962
C	016963
C	016964
C THE ORDER OF THE VARIABLES FOR THE	016965
C ROWS IS	016966
C	016967
C SIZE	016968
C	016969
C PLANT VARIABLES NY	016970
C	016971
C CONTROL VARIABLES NJDELTA	016972
C	016973
C CONTROLLER OUTPUTS NJTU	016974
C	016975
C SENSOR SIGNALS NXSS	016976
C	016977
C	016978
C	016979
C THE COLUMN ORDER IS	016980
C	016981
C PLANT VARIABLES NY	016982
C	016983
C CONTROL VARIABLES NJDELTA	016984
C	016985
C	016986
C	016987
C	016988
C	016989
C	016990
C-----OUTPUT MATRIX -T- (NX,NX)	016991

C			016992
C			016993
C	THIS IS THE SIMILARITY		016994
C	TRANSFORMATION MATRIX		016995
C	THAT INTRODUCES THE (NBTQ + NASS)		016996
C	AUXILIARY VARIABLES INTO THE		016997
C	TRANSFORMED STATE EQUATIONS.		016998
C			016999
C	REF. MATRIX R IN EQ. III-24 OF VOL I		017000
C			017001
C	-----OUTPUT MATRIX Y* (1,NX)		017002
C			017003
C	THIS IS THE TRANSFORMED STATE		017004
C	VECTOR INITIAL CONDITIONS.		017005
C			017006
C	REF. VECTOR Z IN EQ. III-24 OF VOL I		017007
C			017008
C			017009
C	-----OUTPUT MATRIX A* (NX,NX)		017010
C			017011
C	THIS IS THE TRANSFORMED, LINEARIZED		017012
C	STATE VARIABLE COEFFICIENT MATRIX		017013
C	THAT IS THE BASIS FOR THE ENTIRE		017014
C	LINEARIZATION PACKAGE.		017015
C			017016
C	REF. EQ. III-23 OF VOL I		017017
C			017018
C	THE ROW/COL VARIABLE ORDERING AND		017019
C	SIZES ARE:		017020
C			017021
C	VARIABLE ID.	SIZE	017022
C	-----	----	017023
C			017024
C	PLANT VARIABLES	NY2	017025
C			017026
C	PLANT SENSOR SIGNALS	NASS	017027
C			017028
C	CONTROL SYSTEM VARIABLES	ND2	017029
C			017030
C	CONTROL OUTPUTS (B'S)	NBTQ	017031
C			017032
C			017033
C	NOTE- NY2 = NY - NASS		017034
C			017035
C	ND2 = NDELTA - NBTQ		017036
C			017037
C			017038
C			017039
C			017040
C			017041
C			017042
C	-----	RTA	017043
C		RTA*	017044
C	REAL PART	IMAGINARY PART	017045
C			017046
C			017047
C	COMPLEX ROOTS OBTAINED FROM		017048
C	A AND A* RESPECTIVELY.		017049
C			017050
C			017051


```

C                                     THESE ARE THE POLES OF                017052
C                                     THE CLOSED LOOP SYSTEM.                017053
C                                     017054
C                                     017055
C*****                                017056
C                                     017057
C                                     017058
C                                     017059
C      THE FOLLOWING OUTPUTS ARE CHARACTERISTIC OF                017060
C      A SINGLE TRANSFER FUNCTION FREQUENCY RESPONSE.            017061
C                                     017062
C-----OUTPUT MATRIX -AR-                017063
C                                     017064
C      THIS IS THE REDUCED A* MATRIX                017065
C      FOR A PARTICULAR USER SPECIFIED                017066
C      TRANSFER FUNCTION TYPE.                017067
C                                     017068
C      REF. INPUT DATA LIST AND                017069
C      SECTION III. D-2. IN VOLUME I.                017070
C                                     017071
C      THE ROOTS OF AR ARE THE                017072
C      TRANSFER FUNCTION POLES.                017073
C                                     017074
C-----OUTPUT MATRIX COL                017075
C                                     017076
C      THIS IS THE VECTOR (COL) WITH WHICH                017077
C      AR IS AUGMENTED TO DETERMINE THE                017078
C      TRANSFER FUNCTION ZEROS.                017079
C                                     017080
C                                     017081
C                                     017082
C                                     017083
C                                     017084
C                                     017085
C                                     017086
C-----                017087
C      R AR                PART                017088
C      REAL PART  IMAGINARY PART    REAL PART  IMAGINARY PART    017089
C                                     017090
C                                     017091
C                                     017092
C      THESE ARE THE COMPLEX ROOT ARRAYS                017093
C      AS EXTRACTED FROM MATRICES                017094
C      AR AND AR TRANSPOSE RESPECTIVELY.                017095
C                                     017096
C      THE USER SELECTS VIA INPUT WHICH                017097
C      SET OF ROOTS TO USE FOR THE POLES.                017098
C                                     017099
C-----                017100
C      NUM                DEN                017101
C      REAL PART  IMAGINARY PART    REAL PART  IMAGINARY PART    017102
C      .                .                017103
C      .                .                017104
C      .                .                017105
C      .                .                017106
C      .                .                017107
C      NUMERATOR ROOTS                DENOMINATOR ROOTS                017108
C      (TRANSFER FUNCTION ZEROS)    (TRANSFER FUNCTION POLES)        017109
C                                     017110
C                                     017111

```

C	.	.	017112
C	.	.	017113
C	.	.	017114
C			017115
C			017116
C			017117
C			017118
C			017119
C	-----OUTPUT MATRIX ROUTE		017120
C			017121
C	THE REAL PARTS OF THE COMPLEX EIGENVALUES		017122
C	FOR WHICH EIGENVECTORS WILL BE COMPUTED		017123
C			017124
C	-----OUTPUT MATRIX ROUTE		017125
C			017126
C	THE IMAGINARY PARTS OF THE COMPLEX EIGENVALUES		017127
C	FOR WHICH EIGENVECTORS WILL BE COMPUTED		017128
C			017129
C	THE EIGENVALUES FOR WHICH THE ASSOCIATED		017130
C	EIGENVECTORS ARE TO BE COMPUTED CORRESPOND		017131
C	TO THE COMPLEX ROOTS OF THE CHARACTERISTIC		017132
C	MATRIX (OUTPUT MATRIX -AR-)		017133
C			017134
C			017135
C	-----EIGENVALUE =		017136
C			017137
C	-----EIGENVECTOR OF NORMAL MATRIX		017138
C			017139
C	THE SUBROUTINE USED TO COMPUTE EIGENVECTORS		017140
C	ACCEPTS A SINGLE COMPLEX EIGENVALUE (LAMBDA),		017141
C	THE CHARACTERISTIC MATRIX AR AND THE MATRIX AR*AR.		017142
C	IF LAMBDA IS A MULTIPLE ROOT THE ERROR MESSAGE		017143
C	***WARNING*** ZEROS ON DIAGONAL OF FACTORED MATRIX.		017144
C	CHECK FOR MULTIPLE EIGENVALUES		017145
C	IS PRINTED; OTHERWISE		017146
C	THE NA ELEMENTS OF THE EIGENVECTOR XR OF THE NORMAL		017147
C	MATRIX ARE FOUND; THAT IS		017148
C	AR*XR = LAMBDA*XR		017149
C			017150
C			017151
C			017152
C			017153
C			017154
C			017155
C	-----OUTPUT MATRIX RRED		017156
C			017157
C			017158
C	TRANSFER FUNCTION ROOT ARRAY CONTAINING ROOT		017159
C	COUNTS, TIME CONSTANTS, DAMPING, AND FREQUENCY		017160
C	FOR ZEROS AND POLES.		017161
C			017162
C			017163
C	ELE 1 = NO. OF NUMERATOR REAL ROOTS, NNR		017164
C			017165
C	ELE 2 = NO. OF NUMERATOR COMPLEX PAIRS, NNC		017166
C			017167
C	ELE 3 = NO. OF NUMERATOR FREE S'S, NNZ		017168
C			017169
C	ELE 4 = NO. OF DENOMINATOR REAL ROOTS, NDR		017170
C			017171

```

C      ELE 5 = NO. OF DENOMINATOR COMPLEX PAIRS,NDC          017172
C                                                                017173
C      ELE 6 = NO. OF DENOMINATOR FREE S'S,        NDC       017174
C                                                                017175
C      ELE 7 = BUDE GAIN, KB.                        017176
C                                                                017177
C                                                                017178
C      THE ROOTS FOLLOW IN THE ORDER                  017179
C                                                                017180
C                                                                017181
C      ELEMENT LOCATION            DESCRIPTION          017182
C      -----                    -
C                                                                017183
C      8 THRU NNR+7                NUMERATOR TIME CONSTANTS 017184
C                                                                017185
C      NNR+8 THRU 2*NNC + 7        NUMERATOR DAMPING AND FREQUENCIES 017186
C                                                                017187
C                                  ZETA1, OMEGA1, ZETA2, OMEGA2 --- 017188
C                                  FREE S'S ARE NOT INCLUDED    017189
C                                                                017190
C                                                                017191
C      REMAINING ELEMENTS ARE DEMONINATOR ROOTS IN SAME 017192
C      ORDER AS NUMERATOR ROOTS, IE.,                   017193
C                                                                017194
C      TAU(1),...,TAU(NDR),ZETA(1),OMEGA(1),...,ZETA(NDC),OMEGA(NDC) 017195
C                                                                017196
C                                                                017197
C                                                                017198
C                                                                017199
C                                                                017200
C                                                                017201
C                                                                017202
C                                                                017203
C      THE TRANSFER FUNCTION FREQUENCY RESPONSE FOLLOWS 017204
C                                                                017205
C                                                                017206
C----- FREQ/RAD/SEC   FREQ/HERTZ   REAL   IMAG   AMP   DECIBELS   RAD   DEG  017207
C      .               .             .           .           .           .           .  017208
C      .               .             .           .           .           .           .  017209
C      .               .             .           .           .           .           .  017210
C                                                                017211
C                                                                017212
C      THE DAMPED RESONANCES (BASED ON A                 017213
C      POLE OR ZERO) ARE IDENTIFIED                      017214
C      BY ***** IN BOTH THE LEFT AND RIGHT MARGINS.   017215
C                                                                017216
C                                                                017217
C      *****                                           017218
C                                                                017219
C                                                                017220
C      THE FOLLOWING OUTPUTS ARE TYPICAL OF A ROOT      017221
C      LOCUS INVESTIGATION.                             017222
C                                                                017223
C                                                                017224
C-----OUTPUT MATRICES      PDEN                     017225
C                          PNUM                       017226
C                                                                017227
C                          THE DENOMINATOR AND NUMERATOR 017228
C                          TRANSFER FUNCTION POLYNOMIAL 017229
C                          COEFFICIENTS --- ASCENDING POWERS OF S. 017230
C-----                                                    017231

```

C	P(S) =	NUMERATOR	017232
C		AND	017233
C	Q(S) =	DENOMINATOR	017234
C			017235
C		PREPROCESSED	017236
C		POLYNOMIAL COEFFICIENTS AS USED BY RELJCS.	017237
C			017238
C	-----	STARTING POINT = USER IDENTIFIED STARTING POINT	017239
C		FOR THE LOCI.	017240
C			017241
C	-----	SCAN LIMITS = LIMITS ON REAL AND IMAGINARY	017242
C		COMPONENTS FOR THE LOCI.	017243
C			017244
C			017245
C	-----	GAIN ROOTS ERROR	017246
C			017247
C			017248
C			017249
C		ROOT LOCUS OUTPUT	017250
C			017251
C			017252
C			017253
C	*****		017254
C	*****		017255
C		RETURN	017256
C		END	017257

	SUBROUTINE DEFS		017258
C		DEBUG # 116	017259
C			017260
C			017261
C			017262
C		DEFINITIONS OF SYMBOLIC NAMES	017263
C		USED IN DISCOS	017264
C			017265
C			017266
C			017267
C		MAXIMUM SIMULATION MODEL SIZE PARAMETERS	017268
C	-----		017269
C			017270
C		THESE PARAMETERS ARE INTEGER*4 CONSTANTS FOR ANY PARTICULAR	017271
C		COMPILED VERSION OF DISCOS. THEY MAY BE ALTERED ONLY VIA USE	017272
C		OF THE REDIMENSION PROGRAM	017273
C			017274
C		THE FOLLOWING ARE DEFINED IN MAIN, STORED IN /MAXIMUM/ AND PRINTED	017275
C		ALONG WITH THE ECHO OF THE INPUT DATA	017276
C			017277
C		NBYAX = MAXIMUM NUMBER OF BODIES (RIGID + FLEXIBLE)	017278
C		NHMAX = MAXIMUM NUMBER OF HINGE POINTS IN SIMULATION	017279
C		NSFMAX = MAXIMUM NUMBER OF SENSOR POINTS IN SIMULATION	017280
C		NMAMAX = MAXIMUM NUMBER OF MOMENTUM WHEELS IN SIMULATION	017281
C		NMWBCD = MAXIMUM NUMBER OF MOMENTUM WHEELS IN ANY BODY	017282
C		NMCBCD = MAXIMUM NUMBER OF FLEXIBLE BODY MODES ALLOWED PER BODY	017283
C		KMU = MAXIMUM NUMBER OF VELOCITY COORDINATES PER BODY	017284
C		KMU = 6 + NMBCD + NMBCD	017285

```

C      KY      = MAXIMUM NUMBER OF PLANT + CONTROL SYSTEM STATE VARIABLES      017286
C              ALLOWED FOR NON-LINEAR TIME RESPONSE ANALYSIS                    017287
C      KU      = MAXIMUM NUMBER OF VELOCITY COORDINATES IN SIMULATION            017288
C              KU = NCMAX*(G + NMCBOD) + NMWMAX                                  017289
C                                                                                   017290
C      THE FOLLOWING ARE STORED IN OTHER COMMON BLOCKS BUT ARE NOT ECHCED        017291
C      WITH THE INPUT DATA                                                       017292
C                                                                                   017293
C      KSUMRY = MAXO(NHMAX,NSPMAX) STORED IN /SUMMRY/                             017294
C      KR      = MAXIMUM NUMBER OF INDEPENDENT LINEARIZED EQUATIONS WHICH        017295
C              MAY BE SUBJECTED TO FREQUENCY DOMAIN ANALYSIS, STORED            017296
C              IN /LDSIZE/                                                         017297
C      KRT      = 2*KR + 7 (STORED IN /LDSIZE/)                                  017298
C      KRX      = 2*KRT (STORED IN /LDSIZE/)                                      017299
C      KVI      = 2*KRT (STORED IN /LDSIZE/)                                      017300
C      KV2      = KRT (STORED IN /LDSIZE/)                                        017301
C      KVX      = KR (STORED IN /LDSIZE/)                                         017302
C                                                                                   017303
C      THE FOLLOWING ARE NOT STORED IN DISCOS BUT ARE DEFINED IN THE             017304
C      REDIMENSION PROGRAM AND USED THEREIN TO SET DISCOS DIMENSIONS            017305
C                                                                                   017306
C      IGASMX = MAXIMUM NUMBER OF EQUATIONS WHICH MAY BE NUMERICALLY             017307
C              LINEARIZED                                                         017308
C      MAXCNT = MAXIMUM NUMBER OF CONSTANTS WHICH MAY BE INPUTTED BY            017309
C              THE USER IN DYN510                                                017310
C              MAXCNT = KCNT (IN DYN510)                                          017311
C      JMAXC = MAXIMUM NUMBER OF JOINTS (GRID POINTS) ON ANY FLEXIBLE            017312
C              BODY TO BE DESCRIBED BY CONSISTANT MASS MATRIX DATA             017313
C              KJDEF = JMAXC (IN MSMODC)                                          017314
C      JMAXL = MAXIMUM NUMBER OF JOINTS (GRID POINTS) ON ANY FLEXIBLE            017315
C              BODY TO BE DESCRIBED BY LUMPED PARAMETER DATA                  017316
C              KJOINT = JMAXL (IN MSMODL)                                         017317
C      MLTANX = MAXIMUM ALLOWABLE ROW DIMENSION FOR 3 MATRIX IN MULTA           017318
C              MLTAMX (SHOULD BE) = MAXO(6*JMAXC,IGASMX)                       017319
C      MXETAB = MAXIMUM MATRIX DIMENSION IN STABA                               017320
C              MXBTAB (SHOULD BE) = 6*JMAXL                                       017321
C      MLT3MX = MAXIMUM MATRIX DIMENSION IN MULT3                               017322
C      MADMAX = MAXIMUM MATRIX DIMENSION IN MULT4                               017323
C      MAXZF = ROW DIMENSION FOR TIME HISTORY PLOT DATA ARRAY                 017324
C              MAXZP = KRPLT (IN DYN530)                                          017325
C      MXJVPL = SIZE OF TIME HISTORY PLOT VARIABLE SELECTION VECTOR             017326
C              MXJVPL = KCPLT (IN DYN530), DON'T CHANGE                         017327
C      MAXDLM = SIZE OF DUMMY VECTOR DLM IN DYN530                             017328
C      LI      = USED IN REDIMENSION PROGRAM TO SET PARAMETER KR IN            017329
C              DYN540, DEFINED ABOVE                                             017330
C      LS      = MAXIMUM SIZE OF FREQUENCY RESPONSE DATA VECTORS              017331
C              IN /PSTUFF/                                                         017332
C              LS = KDSAVE (IN SFEQ2)                                             017333
C                                                                                   017334
C                                                                                   017335
C                                                                                   017336
C      BASIC SIMULATION MODEL CREATED BY THE USER                             017337
C      -----                                                                    017338
C                                                                                   017339
C      PLANT SIZE: (THESE PARAMETERS ARE INPUTTED VIA DYN510,                   017340
C      ----- ECHCED IN THE STANDARD PRINTOUT, STORED                          017341
C              IN /SPECIF/ AND ARE INTEGER*4 CONSTANTS)                         017342
C                                                                                   017343
C      NB      = TOTAL NUMBER OF RIGID AND FLEXIBLE BODIES                      017344
C              (NB MUST BE .GE.2)                                                017345

```

```

C      NH      = TOTAL NUMBER OF HINGES. HINGE 1 ALWAYS TAKEN TO BE      017346
C      BETWEEN BODY 1 AND THE INERTIAL REFERENCE                        017347
C      NSPT     = TOTAL NUMBER OF SENSOR POINTS. ALL MOMENTUM WHEELS ARE 017348
C      LOCATED AT SENSOR POINTS. ALL POINTS AT WHICH KINEMATIC          017349
C      DATA MUST BE OBTAINED ARE SENSOR POINTS (HINGE POINTS          017350
C      NOT INCLUDED) (NSPT MUST BE .GE.1)                               017351
C      NOFMC    = TOTAL NUMBER OF CONSTANT AND VARIABLE SPEED WHEELS    017352
C                                                                    017353
C      CONTROL SYSTEM SIZE      (THE USER MAY REQUIRE THAT SEVERAL      017354
C      ----- DIFFERENTIAL EQUATIONS BE INTEGRATED                      017355
C      TO DEFINE EFFECTS ACTING ON THE SYSTEM)                          017356
C                                                                    017357
C      NDELTA = TOTAL NUMBER OF USER DEFINED FIRST ORDER DIFFERENTIAL    017358
C      EQUATIONS COULD IN SUBROUTINE CONTROL. AN INTEGER*4             017359
C      CONSTANT STORED IN /SPECIF/ INPUTTED VIA DYN510                  017360
C                                                                    017361
C      PLANT DESCRIPTION : (OBTAINED VIA INPUT DATA CALLS FROM          017362
C      ----- DYN510, ECHOED IN STANDARD PRINTOUT AND                   017363
C      STORED IN EITHER /SPECIF/ OR /NHNS/                               017364
C                                                                    017365
C      ITOPCL = PLANT TOPOLOGY. AN INTEGER*4 ARRAY DIMENSIONED (2,NHMAX) 017366
C      USED ALSO TO DEFINE POSITIVE DIRECTION OF RELATIVE MOTION        017367
C      AT HINGE POINT J, J=1,2,....,NH                                  017368
C      BODY NUMBER N=ITOPCL(1,J), WHICH CONTAINS THE Q-FRAME,           017369
C      IS CONTIGUOUS TO                                                 017370
C      BODY NUMBER M=ITOPCL(2,J), WHICH CONTAINS THE P-FRAME,           017371
C      BY DEFINITION:                                                  017372
C      HINGE POINT 1 IS BODY 1 REFERENCE POINT                         017373
C      FOR J=1, N=1 AND M=0; ALSO,                                     017374
C      THE MOTION OF THE Q-FRAME WILL BE MEASURED                      017375
C      RELATIVE TO THE P-FRAME AT EVERY HINGE                          017376
C      POINT IN THE SIMULATION                                          017377
C      IRGFLX = BODY TYPE, AN INTEGER*4 ARRAY DIMENSIONED (NBMAX)       017378
C      IRGFLX(N) = 0 IMPLIES THAT BODY N IS A RIGID BODY               017379
C      IRGFLX(N) = M IMPLIES THAT BODY N IS A FLEXIBLE BODY            017380
C      AND THAT M MODES OF VIBRATION ARE TO BE                         017381
C      USED TO COMPUTE ITS FLEXURAL DEFORMATION                        017382
C      IMD      = MOMENTUM WHEELS, INTEGER*4 ARRAY DIMENSIONED (3,NM*MAX) 017383
C      FOR MOMENTUM WHEEL NUMBER J, J=1,2,....,NOFMC                  017384
C      IMD(1,J) = SENSOR POINT NUMBER ASSOCIATE WITH WHEEL J           017385
C      IMD(2,J) = THE AXIS NUMBER (1,2 OR 3) OF THE SENSOR             017386
C      POINT REFERENCE FRAME WHICH IS ALIGNED WITH                    017387
C      SPIN AXIS OF THE WHEEL J                                         017388
C      IMD(3,J) = 1 IF WHEEL J IS ACTIVE OR VARIABLE SPEED            017389
C      0 IF WHEEL J IS CONSTANT SPEED                                  017390
C      NMOW      = MOMENTUM WHEEL/BODY TABLE, COMPUTED INTEGER*4 ARRAY 017391
C      DIMENSIONED (2+NH*BCD,NBMAX), FOR BODY N, N=1,2,....,NB       017392
C      NMOW(1,N) = NUMBER OF MOMENTUM WHEELS ON BODY N                 017393
C      NMOW(2,N) = NUMBER OF VARIABLE SPEED WHEELS ON BODY N           017394
C      NMOW(3,N) = LOWEST MAGNITUDE WHEEL NUMBER ON BODY N             017395
C      .          = NEXT TO LOWEST WHEEL NUMBER ON BODY N             017396
C      .          = ETC.                                                017397
C      .          =                                                    017398
C      NMOW(L,N) = HIGHEST MAGNITUDE WHEEL NUMBER ON BODY N           017399
C      WHERE L=2+NMOW(1,N)                                              017400
C      IFISMA = SENSOR POINT/BODY TABLE, INPUTTED INTEGER*4 ARRAY     017401
C      DIMENSIONED (NSP*MAX)                                           017402
C      IFISMA(J) = BODY NUMBER OF THE BODY ON WHICH                   017403
C      SENSOR POINT J IS LOCATED J=1,2,....,NSPT                      017404
C      NMPUI   = HINGE/BODY TABLE, COMPUTED INTEGER*4 ARRAY           017405

```

C	DIMENSIONED (NBMAX)	017406
C	NHPDI(N) = NUMBER OF HINGE POINTS ON BODY N,N=1,....,NB	017407
C	NSPOI = SENSOR/BODY TABLE, COMPUTED INTEGER*4 ARRAY	017408
C	DIMENSIONED (NBMAX)	017409
C	NSPOI(N) = NUMBER OF SENSOR POINTS ON BODY N,N=1,....,NB	017410
C	IHDATA = HINGE POINT CONSTRAINT AND EULER ANGLE TYPE, INPUTTED	017411
C	INTEGER*4 ARRAY DIMENSIONED (7,NHMAX)	017412
C	FOR HINGE POINT J, J=1,2,....,N1	017413
C	IHDATA(1,J) = ITYPE (CODE NUMBER ASSIGNED TO THE DESIRED	017414
C	EULER ANGLE ROTATION SEQUENCE TO BE USED	017415
C	TO ORIENT Q-FRAME WRT P-FRAME); THAT IS	017416
C	ITYPE =1-(1,2,3) ITYPE = 7-(2,1,2)	017417
C	ITYPE =2-(1,2,1) ITYPE = 8-(2,1,3)	017418
C	ITYPE =3-(1,3,1) ITYPE = 9-(3,1,2)	017419
C	ITYPE =4-(1,3,2) ITYPE = 10-(3,1,3)	017420
C	ITYPE =5-(2,3,1) ITYPE = 11-(3,2,3)	017421
C	ITYPE =6-(2,3,2) ITYPE = 12-(3,2,1)	017422
C	IHDATA(2,J) = CODE TO DEFINE THETA 1 ROTATION CONSTRAINT	017423
C	IHDATA(3,J) = CODE TO DEFINE THETA 2 ROTATION CONSTRAINT	017424
C	IHDATA(4,J) = CODE TO DEFINE THETA 3 ROTATION CONSTRAINT	017425
C	IHDATA(5,J) = CODE TO DEFINE X TRANSLATION CONSTRAINT	017426
C	IHDATA(6,J) = CODE TO DEFINE Y TRANSLATION CONSTRAINT	017427
C	IHDATA(7,J) = CODE TO DEFINE Z TRANSLATION CONSTRAINT	017428
C	CONSTRAINT CODE = 0 - NO CONSTRAINT	017429
C	CONSTRAINT CODE = 1 - FIXED CONSTRAINT, NO RELATIVE	017430
C	MOTION ALLOWED IN THIS DIRECTION	017431
C	CONSTRAINT CODE = 2 - RHEONOMIC CONSTRAINT, RELATIVE	017432
C	MOTION IN THIS DIRECTION TO BE	017433
C	USER DEFINED VIA SUBROUTINES	017434
C	ADT AND ADUT	017435
C		017436
C	LINEAR OR NON-LINEAR ANALYSIS	017437
C	-----	017438
C	IFLNER = INPUT PARAMETER IPDATA(3) WHICH IS READ IN, IN DYN10	017439
C	AND STORED IN /ILINER/	017440
C	IFLNER = 0 IF THE USER DESIRES THE COUPLED NON-LINEAR	017441
C	EQUATIONS OF MOTION TO BE INTEGRATED WITH	017442
C	RESULTANT TIME RESPONSE OUTPUT	017443
C	IFLNER = 1 IF THE USER DESIRES THE COUPLED NON-LINEAR	017444
C	EQUATIONS OF MOTION TO BE NUMERICALLY	017445
C	LINEARIZED AND EITHER THE LINEAR TIME OR	017446
C	FREQUENCY RESPONSE ANALYSIS PERFORMED	017447
C		017448
C	STABILITY ANALYSIS: (TRANSFER FUNCTIONS MAY BE OBTAINED	017449
C	----- WHICH DEFINE THE FREQUENCY RESPONSE	017450
C	CHARACTERISTICS OF ANY USER DEFINED	017451
C	OUTPUT 'TORQUE SIGNAL' TO ANY USER	017452
C	DEFINED INPUT 'SENSOR SIGNAL')	017453
C		017454
C	NXSS = TOTAL NUMBER OF USER DEFINED SENSOR SIGNALS. THESE ARE	017455
C	ASSUMED TO BE FUNCTIONS OF THE PLANT STATE VARIABLES AND	017456
C	ARE NUMERICALLY LINEARIZED WITH RESPECT TO THEM. THIS	017457
C	PARAMETER IS USER DEFINED IN CONTRL AND STORED	017458
C	IN /LDSIZE/	017459
C	NBTQ = TOTAL NUMBER OF USER DEFINED TORQUE SIGNALS. THESE ARE	017460
C	ASSUMED TO BE FUNCTIONS OF THE CONTROL SYSTEM STATE	017461
C	VARIABLES AND ARE NUMERICALLY LINEARIZED WITH RESPECT TO	017462
C	THEM. THIS PARAMETER IS USER DEFINED IN CONTRL AND STORED	017463
C	IN /LDSIZE/	017464
C	NAUX = TOTAL NUMBER OF AUXILIARY EQUATIONS, DEFINED IN EQADD	017465

C	AND STORED IN /DUNAUX/	017466
C	NAUX = NXSS + NBTD	017467
C	NDELTA = DEFINED EQUAL TO NDELTA IN CONTRL, STORED IN /LOSIZE/	017468
C		017469
C		017470
C		017471
C	DISCOS COMPUTATION CONTROL PARAMETERS	017472
C	-----	017473
C		017474
C	STATE VECTOR, SIZE AND ARRANGEMENT OF STATE VARIABLES	017475
C	-----	017476
C	THE FOLLOWING PARAMETERS ARE INTEGER*4 CONSTANTS, COMPUTED	017477
C	IN DYN510 AND STORED IN /SPECIF/. THEY ARE USED TO DEFINE THE	017478
C	NUMBER OF AND LOCATION OF VARIOUS COMPONENTS OF THE TOTAL	017479
C	SYSTEM STATE VECTOR	017480
C	NBETA = TOTAL NUMBER OF (BETA) COORDINATES IN THE STATE VECTOR,	017481
C	THESE COORDINATES DEFINE THE TRANSLATION OR ROTATIONAL	017482
C	MOTION ASSOCIATE WITH EACH UNCONSTRAINED OR RHECNOMICALLY	017483
C	CONSTRAINED DEGREE OF FREEDOM	017484
C	NLAM = TOTAL NUMBER OF (LAMBDA) CONSTRAINT EQUATIONS. A (LAMBDA)	017485
C	CONSTRAINT EQUATION IS DEFINED FOR EACH FIXED OR	017486
C	RHECNOMICALLY CONSTRAINED DEGREE OF FREEDOM	017487
C	NU = TOTAL NUMBER OF (U'S) VELOCITY COORDINATES IN THE STATE	017488
C	VECTOR. THE BREAK-DOWN OF VELOCITY COORDINATES FOR BODY N	017489
C	IS AS FOLLOWS:	017490
C	3 - ROTATION COORDINATES (COMPONENTS OF THE INERTIAL	017491
C	ANGULAR VELOCITY VECTOR WRT THE BODY N REFERENCE)	017492
C	3 - TRANSLATION COORDINATES (COMPONENTS OF THE INERTIAL	017493
C	LINEAR VELOCITY VECTOR WRT THE BODY N REFERENCE)	017494
C	IRGFLX(N) - GENERALIZED MODAL VELOCITY COORDINATES ASSOCIATED	017495
C	WITH FLEXIBLE BODY N DEFORMATION. ZERO IF BODY N	017496
C	IS A RIGID BODY	017497
C	NACW(2,N) - WHEEL SPIN RATE COORDINATES ASSOCIATED WITH THE	017498
C	ACTIVE OR VARIABLE SPEED WHEELS IN BODY N	017499
C	ZERO IF BODY N HAS NO EMBEDDED ACTIVE WHEELS	017500
C	NEQ = TOTAL NUMBER OF STATE EQUATIONS WHICH ARE SETUP TO	017501
C	DEFINE TOTAL SYSTEM TIME RESPONSE. IN DYN510	017502
C		017503
C	NEQ = IRGFLX(1) + ... + IRGFLX(NB)	017504
C	+ NU + NBETA + NDELTA	017505
C		017506
C	LENU = STATE VECTOR LENGTH TABLE, COMPUTED INTEGER*4 ARRAY	017507
C	DIMENSIONED (2*NBMAX+2)	017508
C	LENU(N) = NUMBER OF VELOCITY COORDINATES (U'S)	017509
C	ASSOCIATED WITH BODY N. N=1,2,...,NB	017510
C	LENU(N+NB) = NUMBER OF MODAL DISPLACEMENT COORDINATES	017511
C	ASSOCIATED WITH BODY N. N=1,2,...,NB	017512
C	LENU(2*NB+1) = NBETA	017513
C	LENU(2*NB+2) = NDELTA	017514
C	LOCU = STATE VECTOR LOCATION TABLE, COMPUTED INTEGER*4 ARRAY	017515
C	DIMENSIONED (2*NBMAX+2)	017516
C	LOCU(N) = LOCATION IN STATE VECTOR OF FIRST VELOCITY	017517
C	COORDINATE ASSOCIATED WITH BODY N. N=1,...,NB	017518
C	LOCU(N+NB) = LOCATION IN STATE VECTOR OF FIRST MODAL	017519
C	DISPLACEMENT COORDINATE ASSOCIATED WITH	017520
C	BODY N. N=1,2,...,NB	017521
C	LOCU(2*NB+1) = LOCATION IN STATE VECTOR OF FIRST (BETA)	017522
C	COORDINATE USED TO DEFINE CONTIGUOUS BODY	017523
C	RELATIVE MOTION	017524
C	LOCU(2*NB+2) = LOCATION IN STATE VECTOR OF FIRST STATE	017525


```

C                                     VARIABLE ASSOCIATED WITH THE NDELTA USER      017526
C                                     DEFINED CONTROL EQUATIONS                      017527
C                                     017528
C THE FOLLOWING PARAMETERS ARE INTEGER*4 CONSTANTS, COMPUTED                      017529
C INTERNALLY AND STORED IN /LDSIZE/. THEY ARE USED IN THE SETUP                  017530
C OF THE LINEARIZED EQUATIONS OF MOTION                                         017531
C NX      = NUMBER OF INDEPENDENT COORDINATES IN THE STATE VECTOR Y,             017532
C          THIS IS COMPUTED IN LINEAR                                             017533
C NJC      = NUMBER OF INDEPENDENT COORDINATES IN THE STATE VECTOR Y             017534
C          PLUS THE NUMBER OF USER DEFINED SENSOR AND TORQUE SIGNALS              017535
C          (AUXILIARY EQUATIONS) CODED IN EQADD; COMPUTED IN LINEAR              017536
C          NJC = NX + NAUX                                                         017537
C NY2      = NUMBER OF INDEPENDENT PLANT COORDINANTS IN THE STATE                017538
C          VECTOR MINUS THE NUMBER OF USER DEFINED SENSOR SIGNALS                017539
C          NY2 = NX - NDELTA - NXSS                                               017540
C          COMPUTED IN DYN540                                                     017541
C ND2      = NUMBER OF INDEPENDENT CONTROL SYSTEM COORDINATES IN THE            017542
C          STATE VECTOR MINUS THE NUMBER OF USER DEFINED TORQUE                  017543
C          SIGNALS                                                                017544
C          ND2 = NDELTA - NBTQ                                                    017545
C          COMPUTED IN DYN540                                                     017546
C                                     017547
C          INTEGRATION CONTROL                                                    017548
C          -----                                                                017549
C JIL      = 1,2,3 OR 4; DEFINES THE CYCLE NUMBER WITHIN THE FOUR STEP          017550
C          RUNGE KUTTA INTEGRATION ROUTINE THAT COMPUTATION IS                   017551
C          CURRENTLY AT; UPDATED IN RKADAM AND STORED IN /JILFLG/.               017552
C NOTE     ADAMS PREDICTOR/CORRECTOR INTEGRATION MAY BE USED BY                 017553
C ----- SETTING ITYPE=2 IN THE DATA STATEMENT IN RKADAM, IF SO               017554
C          JIL WILL ALWAYS BE EQUAL TO 4 AFTER THE FIRST RUNGE                   017555
C          KUTTA STARTING STEP.                                                  017556
C                                     017557
C THE FOLLOWING ARE REAL*8 PARAMETERS STORED IN /TIMESS/ AND USED                 017558
C TO DEFINE DIFFERENT TIME PARAMETERS                                           017559
C ST       = STARTT = INPUT PARAMETER TMDATA(1) READ IN, IN DYN510;             017560
C          USED TO DEFINE THE STARTING TIME FOR TIME                            017561
C          HISTORY SIMULATION                                                    017562
C DT       = DELTAT = INPUT PARAMETER TMDATA(2) READ IN, IN DYN510;            017563
C          USED TO DEFINE THE INTEGRATION STEP SIZE                             017564
C ET       = ENDT   = INPUT PARAMETER TMDATA(3) READ IN, IN DYN510;            017565
C          USED TO DEFINE THE END OF SIMULATION TIME                            017566
C T        = TIME PARAMETER, INCREMENTED IN RKADAM, COMPUTATION                 017567
C          STARTS AT T=ST AND IS STOPPED AT T=ET                                017568
C TMST     = TIME ELAPSED IN SIMULATION FROM T=ST                              017569
C                                     017570
C THE FOLLOWING ARE VARIABLES INITIALIZED IN DYN520, UPDATED IN                  017571
C RKADAM AND STORED IN /QPRKTA/                                                 017572
C GRK      = REAL*8 WORK AREA FOR RUNGE-KUTTA INTEGRATION ROUTINE               017573
C          DIMENSIONED (KY)                                                      017574
C FRK      = REAL*8 ARRAY OF CONSTANTS REQUIRED FOR RUNGE KUTTA                 017575
C          INTEGRATION ROUTINE DIMENSIONED (4)                                  017576
C NT       = INTEGER*4 VARIABLE USED AS A COUNTER. ITS STORED VALUE IS           017577
C          EQUAL TO THE TOTAL NUMBER OF INTEGRATION STEPS TAKEN IN              017578
C          NON-LINEAR TIME RESPONSE ANALYSIS FROM TIME STARTT TO                017579
C          INSTANTANEOUS TIME T.                                                 017580
C                                     017581
C          OUTPUT DATA CONTROL                                                  017582
C          -----                                                                017583
C NOPRINT  = INPUT PARAMETER IPDATA(1) READ IN, IN DYN510 AND STORED           017584
C          IN /MISCNO/. PRINT OUTPUT DATA EVERY NOPRINT MULTIPLES              017585

```

```

C      OF THE INTEGRATION STEP                                C17586
C      NOPLCT = INPUT PARAMETER IPDATA(2) READ IN, IN DYSNIO AND STORED C17587
C      IN /MISCND/. PLOT OUTPUT DATA EVERY NOPLCT MULTIPLES C17588
C      OF THE INTEGRATION STEP                                C17589
C      NRPLCT = BOTH ARE INTEGERS COMPUTED IN PLUTAR, LPLTAR AND DYSN30 C17590
C      NOPLCT AND STORED IN /PLTDIA/ THEY ARE REQUIRED FOR SC-4020 C17591
C      PLOT GENERATION C17592
C      KSAVE = TOTAL NUMBER OF DATA POINTS TO BE SAVED IN THE /PSTUFF/ C17593
C      ARRAYS, WHICH ARE EXCLUSIVELY RESERVED FOR THE STORAGE OF C17594
C      FREQUENCY RESPONSE DATA COMPUTED IN JFREQ2 C17595
C C17596
C C17597
C      MODE DEPENDENT PARAMETERS C17598
C----- C17599
C C17600
C      MASS MATRIX WITH ASSOCIATED MODE DEPENDENT PARAMETERS C17601
C----- C17602
C C17603
C      AMU = MASS MATRIX ARRAY, REAL*8 TIME VARYING PARAMETERS STORED C17604
C      IN /AMUBW/. DIMENSIONED(KMU,KMU,NBMAX) C17605
C      UPON ENTRY INTO DYSN20: C17606
C      ((AMU(I,J,N),I=1,LV),J=1,LV) = MASS MATRIX FOR BODY N C17607
C      IN ITS UNDEFORMED STATE WHERE C17608
C      LV = 6 + IRGFLX(N) + NMOW(2,N) C17609
C      THEREAFTER: C17610
C      ((AMU(I,J,N),I=1,LV),J=1,LV) = MASS MATRIX FOR BODY N C17611
C      AT TIME T IN THE DEFORMED STATE C17612
C      (COMPUTED IN MOEN), WHERE C17613
C      LV = 6 + IRGFLX(N) + NMOW(2,N) C17614
C      DETAILS SEE EQUATIONS 11-39 AND 86 IN VOL 1: C17615
C      ((AMU(I,J,N),I=1,3),J=1,3) = INERTIA TENSOR FOR BODY N C17616
C      RELATIVE TO BODY N REF POINT WRT C17617
C      BODY N REFERENCE FRAME C17618
C      ((AMU(I,J,N),I=1,3),J=4,6) = SKEW TENSOR FORM OF THE BODY N C17619
C      MASS MOMENT VECTOR RELATIVE TO C17620
C      BODY N REF POINT WRT BODY N C17621
C      REFERENCE FRAME C17622
C      ((AMU(I,J,N),I=4,6),J=4,6) = DIAGONAL MASS MATRIX, TOTAL MASS C17623
C      OF BODY N ON DIAGONAL, ZERO OFF C17624
C      THE DIAGONAL C17625
C      FOR THE FOLLOWING LET: C17626
C      LE = 6+IRGFLX(N) C17627
C      LV = 6+IRGFLX(N)+NMOW(2,N) C17628
C      ((AMU(I,J,N),I=1,3),J=7,LE) = INERTIAL COUPLING TERMS BETWEEN C17629
C      ANGULAR AND MODAL ACCELERATION C17630
C      COORDINATES FOR BODY N C17631
C      ((AMU(I,J,N),I=4,6),J=7,LE) = INERTIAL COUPLING TERMS BETWEEN C17632
C      LINEAR AND MODAL ACCELERATION C17633
C      COORDINATES FOR BODY N C17634
C      ((AMU(I,J,N),I=7,LE),J=7,LE) = INERTIAL COUPLING TERMS BETWEEN C17635
C      MODAL ACCELERATION COORDINATES C17636
C      FOR BODY N (DIAGONAL FOR C17637
C      ORTHOGONAL MODES) C17638
C      ((AMU(I,J,N),I=1,LE),J=LE+1,LV) = INERTIAL COUPLING TERMS C17639
C      BETWEEN BODY N ACCEL. C17640
C      CORD. AND WHEEL ACCEL. C17641
C      ((AMU(I,J,N),I=LE+1,LV),J=LE+1,LV) = INERTIAL COUPLING BETWEEN C17642
C      VARIABLE SPEED WHEELS ON C17643
C      BODY N C17644
C      IN SUBROUTINE YOUT THE MASS MATRIX INVERSE IS NEEDED. IT C17645

```

```

C      IS COMPUTED IN INVNP AND STORED IN THE AMU ARRAY      017646
C      THUS DESTROYING ALL MASS MATRIX DATA DEFINED BELOW  017647
C      ** NOTE IT MAY BE RECOVERED BY REINVERTING AMU ARRAY  017648
C                                                                017649
C      THE FOLLOWING PARAMETERS ARE REAL*8 ARRAYS STORED IN /INTGRL/. 017650
C      THEIR ELEMENT MAGNITUDES ARE EVALUATED IN EITHER MRIGID, MSMODC 017651
C      OR MSMOUL AND STORED ON NTAPE1 (TO CONSERVE COMPUTE STORAGE). 017652
C      UPON ENTRY INTO DYN20 A READ(NTAPE1) INSTRUCTION IS EXECUTED 017653
C      AND THE APPROPRIATE TIME INVARIANT ARRAYS LOADED.      017654
C      THESE ARRAY WILL BE USED IN MGEN TO COMPUTE THE ELEMENTS OF THE 017655
C      TIME VARYING MASS MATRIX DEFINED BY EQUATION 11-86, VOL 1  017656
C      AM      = UNDEFORMED MASS MATRIX ARRAY (M SUB 0) IN 11-86) 017657
C               DIMENSIONED (((6+NMDBOD)*(7+NMDBOD))/2,NBMAX). UPON ENTRY 017658
C               INTO DYN20 IT IS SET BY THE FOLLOWING CODE:      017659
C               DO 3 N=1,NB                                       017660
C               NXE = IRGFLX(N)                                     017661
C               NP6 = 6+NXE                                         017662
C               KNT = 0                                             017663
C               DO 6 I=1,NP6                                         017664
C               DO 6 J=1,NP6                                         017665
C               KNT = KNT+1                                         017666
C               6 AM(KNT,N) = AMU(I,J,N)                           017667
C               3 CONTINUE                                          017668
C      SINCE THE MASS MATRIX IS SYMMETRIC                        017669
C      ONLY THE UPPER TRIANGULAR PORTION NEED BE SAVED          017670
C      (AM(I,N),I=1,(NP6*(NP6+1))/2) = UNDEFORMED MASS      017671
C      MATRIX FOR BODY N                                         017672
C      ACOF      = COEFFICIENTS WHICH DEFINE THE LINEAR DEPENDENCE OF THE 017673
C      MASS MOMENT ON FLEXURAL DEFORMATION.                     017674
C      DIMENSIONED (9,NMDBOD,NBMAX). THAT IS:                  017675
C      (ACOF(I,M,N),I=1,9) = 9 ALPHA COEFFICIENTS IN EQ 11-88 017676
C      WHICH DEFINE THE LINEAR DEPENDENCE                       017677
C      OF BODY N MASS MOMENT ON BODY N                           017678
C      MODE M DEFORMATION.                                       017679
C      ECCF      = COEFFICIENTS WHICH DEFINE THE LINEAR DEPENDENCE OF THE 017680
C      INERTIA TENSOR ON FLEXURAL DEFORMATION.                  017681
C      DIMENSIONED (6,NMDBOD,NBMAX). THAT IS:                  017682
C      (ECCF(I,M,N),I=1,6) = 6 LOWER CASE B COEFFICIENTS IN  017683
C      EQ 11-88 WHICH DEFINE THE LINEAR                         017684
C      DEPENDENCE OF BODY N INERTIA TENSOR                      017685
C      ON BODY N MODE M DEFORMATION                             017686
C      COFYZ      = COEFFICIENTS WHICH DEFINE THE LINEAR DEPENDENCE OF THE 017687
C      COFX2      = INERTIAL COUPLING TERMS BETWEEN ANGULAR AND MODAL 017688
C      COFX1      = ACCELERATION COEFFICIENTS, EACH OF THE THREE ARRAYS ARE 017689
C      DIMENSIONED (NMDBOD,NMDBOD,NBMAX)                       017690
C      (COFYZ(I,M,N),I=1,IRGFLX(N))= IRGFLX(N) C SUB YZ COUPLING 017691
C      COEFFICIENTS IN EQ 11-88. DEFINES                        017692
C      THE LINEAR DEPENDENCE OF BODY N                          017693
C      X-AXIS ROTATION - MODE M COUPLING                        017694
C      COEFFICIENTS ON MODE 1,I=1,....                          017695
C      IRGFLX(N) DEFORMATION                                     017696
C      (COFX2(I,M,N),I=1,IRGFLX(N))= IRGFLX(N) C SUB XZ COUPLING 017697
C      COEFFICIENTS IN EQ 11-88. COUPLING                       017698
C      Y-AXIS ROTATION - MODE M BODY N                         017699
C      (COFX1(I,M,N),I=1,IRGFLX(N))= IRGFLX(N) C SUB XY COUPLING 017700
C      COEFFICIENTS IN EQ 11-88. COUPLING                       017701
C      Z-AXIS ROTATION - MODE M BODY N                         017702
C      COF11      = COEFFICIENTS WHICH DEFINE THE QUADRATIC DEPENDENCE OF THE 017703
C      COF12      = INERTIA TENSOR ON FLEXURAL DEFORMATION.    017704
C      COF13      = DIMENSIONED (NMDBOD,NMDBOD,NBMAX)          017705

```

```

C      CCF21  CCF11(1,J,N) = COEFFICIENTS WHICH DEFINE THE QUADRATIC      017706
C      CCF22  CCF22(1,J,N)  DEPENDENCE OF PRINCIPAL AX, YY, ZZ MOMENTS    017707
C      CCF32  CCF32(1,J,N)  OF INERTIA RESP. OF BODY N ON MODE I AND      017708
C                                          MODE J FLEXURAL DEFORMATION, SEE EQ 11-89  017709
C      CCF12(1,J,N) = COEFFICIENTS WHICH DEFINE THE QUADRATIC      017710
C      CCF13(1,J,N)  DEPENDENCE OF PRODUCTS XY, XZ, YZ                017711
C      CCF23(1,J,N)  OF INERTIA RESP. OF BODY N ON MODE I AND      017712
C                                          MODE J FLEXURAL DEFORMATION, SEE EQ 11-89  017713
C                                          017714
C      MODAL COMPONENTS AT HINGE AND SENSOR POINTS                    017715
C      -----                                                        017716
C      THE FOLLOWING PARAMETERS ARE REAL*8 ARRAYS STORED IN /HANDS/ THE  017717
C      ACTUAL DATA TO BE LOADED INTO THESE ARRAYS IS INPUTTED EITHER    017718
C      VIA MSMODL OR MSMODC, IT IS STORED TEMPORARILY ON NTAPE1 THEN      017719
C      UNLOADED IN DYN520 INTO THE APPROPRIATE ARRAYS.                  017720
C      PATH    = HINGE POINT MODAL DISPLACEMENT ARRAY,                017721
C                DIMENSIONED (3,NMODBD,2*NHMAX-1).                    017722
C      SIGH    = HINGE POINT MODAL ROTATION ARRAY,                     017723
C                DIMENSIONED (3,NMODBD,2*NHMAX-1)                     017724
C                AT HINGE POINT NCH, NCH=2,3,...,NH                    017725
C                THE Q-FRAME IS CONTAINED IN BODY N=ITOPUL(1,NCH) AND    017726
C                THE P-FRAME IS CONTAINED IN BODY M=ITOPUL(2,NCH)        017727
C      (PATH(I,NN,2*NCH-3),I=1,3) = 3 DISPLACEMENT COMPONENTS OF MODE  017728
C                NN AT THE Q-FRAME ORIGIN WRT THE                       017729
C                BODY N FIXED REFERENCE                                017730
C      (PATH(I,MM,2*NCH-2),I=1,3) = 3 DISPLACEMENT COMPONENTS OF MODE  017731
C                MM AT THE P-FRAME ORIGIN WRT THE                       017732
C                BODY M FIXED REFERENCE                                017733
C      (SIGH(I,NN,2*NCH-3),I=1,3) = 3 ROTATION COMPONENTS OF MODE      017734
C                NN AT THE Q-FRAME ORIGIN WRT THE                       017735
C                BODY N FIXED REFERENCE                                017736
C      (SIGH(I,MM,2*NCH-2),I=1,3) = 3 ROTATION COMPONENTS OF MODE      017737
C                MM AT THE P-FRAME ORIGIN WRT THE                       017738
C                BODY M FIXED REFERENCE                                017739
C      PATHS   = SENSOR POINT MODAL DISPLACEMENT ARRAY,                017740
C                DIMENSIONED (3,NMODBD,NSPMAX)                        017741
C      SIGS    = SENSOR POINT MODAL ROTATION ARRAY                      017742
C                DIMENSIONED (3,NMODBD,NSPMAX)                        017743
C                AT SENSOR POINT NOS WHICH IS ON BODY N=IFTSMW(NOS)     017744
C      (PATHS(I,NN,NOS),I=1,3) = 3 DISPLACEMENT COMPONENTS OF MODE NN   017745
C                AT SENSOR POINT NOS WRT BODY N REF                    017746
C      (SIGS(I,NN,NOS),I=1,3) = 3 ROTATION COMPONENTS OF MODE NN        017747
C                AT SENSOR POINT NOS WRT BODY N REF                    017748
C                                          017749
C      MODAL STIFFNESS AND DAMPING MATRICES                          017750
C      -----                                                        017751
C      THE FOLLOWING PARAMETERS ARE REAL*8 ARRAYS STORED IN /INTGRL/,    017753
C      THESE ARRAYS ARE LOADED IN DYN520 BY A READ(NTAPE1) STATEMENT.    017754
C      THE DATA ON NTAPE1 WAS GENERATED IN AND BY CALLS FROM THE        017755
C      FLEXIBLE BODY INPUT ROUTINES MSMODL AND MSMODC.                  017756
C      AK      = MODAL STIFFNESS MATRIX, DIMENSIONED (NMODBD,NMODCD,NEMAX) 017757
C      LET:                                          017758
C      NM=IRGFLX(N) = NUMBER OF MODES USED FOR BODY N.                  017759
C      THEN                                          017760
C      ((AK(I,J,N),I=1,NM),J=1,NM) = NM BY NM MODAL STIFFNESS          017761
C                MATRIX FOR BODY N. NON-ORTHOGONAL MODES                017762
C                MAY BE USED; IF SO, THIS MATRIX WILL BE                 017763
C                SYMMETRIC BUT NON-DIAGONAL.                            017764
C      FOR ORTHOGONAL MODES:                                          017765

```

```

C          AK(M,M,N) = (GENERALIZED MASS OF MODE M BODY N) * (MODE M 017766
C                          NATURAL FREQUENCY)**2 017767
C      AD      = MODAL DAMPING MATRIX, DIMENSIONED (NMDBOD,NMDOCD,NHMAX) 017768
C      LET: 017769
C          NM = IROFLX(N) = NUMBER OF MODES USED FOR BODY N 017770
C      THEN 017771
C          ((AD(I,J,N),I=1,NM),J=1,NM) = NM BY NM MODAL DAMPING 017772
C                          MATRIX FOR BODY N. IF THE DAMPING MATRIX 017773
C                          IS NOT PROPORTIONAL TO A LINEAR FUNCTION 017774
C                          OF THE MASS AND STIFFNESS MATRICES THIS 017775
C                          MODAL DAMPING MATRIX WILL BE SYMMETRIC AND 017776
C                          NON-DIAGONAL 017777
C      FOR PROPORTIONAL DAMPING IN BODY N: 017778
C          AD(M,M,N) = 2 * (GENERALIZED MASS OF MODE M) 017779
C                      * (DAMPING ZETA FOR MODE M) 017780
C                      * (NATURAL FREQUENCY OF MODE M) 017781
C 017782
C 017783
C      GRAVITY GRADIENT DATA, INPUTTED AND COMPUTED 017784
C      ----- 017785
C 017786
C      WV      = GRAVITY GRADIENT INPUT DATA ARRAY. THESE REAL*8 CONSTANTS 017787
C                ARE USED TO COMPUTE DATA TO BE STORED IN /GGDATA/ 017788
C      (WV(I),I=1,3) = 3 COMPONENTS OF THE GRAVITY VECTOR WRT 017789
C                INERTIAL REFERENCE (UNITS OF ACC) 017790
C      WV(4)      = DISTANCE FROM GRAVITY SOURCE TO VICINITY 017791
C                OF BODY CLUSTER CENTER OF MASS 017792
C 017793
C      THE FOLLOWING PARAMETERS ARE REAL*8 VARIABLES STORED IN /GGDATA/ 017794
C      THE USER MAY, IN CONTRL, DEFINE THESE PARAMETERS TO BE TIME 017795
C      FUNCTIONS SO THAT MOTION IN ORBIT MAY BE SIMULATED 017796
C      RCMAG = WV(4) DISTANCE FROM GRAVITY SOURCE TO VICINITY OF SYSTEM 017797
C      CMAG  = RESULTANT GRAVITATIONAL ACCELERATION 017798
C            GMAG = SQRT(WV(1)**2 + WV(2)**2 + WV(3)**2) 017799
C      CMAGI = COMPONENTS OF A UNIT VECTOR FROM THE GRAVITY SOURCE TO 018000
C            THE VICINITY OF THE SYSTEM CENTER OF MASS WRT INERTIAL 018001
C            REFERENCE, DIMENSIONED (3). FROM JYNS10 018002
C            CMAGI(I) = WV(I)/GMAG; I=1,2,3 018003
C            (NOT COMPUTE IF (GMAG.EQ.0)) 018004
C      GGS    = GRAVITY GRADIENT EFFECTS ON ELASTIC COORDINATES, A REAL*8 018005
C                ARRAY DIMENSIONED (NMDBOD,9,NHMAX) AND STORED IN /GCSAVE/ 018006
C                THE ELEMENTS OF THIS ARRAY ARE COMPUTED IN MGEN BASED 018007
C                UPON THEORY ON PAGES 48-52, VOL 1. THEY ARE 018008
C                DEFORMATION DEPENDENT TERMS REQUIRED FOR COMPUTATION 018009
C                OF GRAVITY GRADIENT FORCES AND TORQUES 018010
C                FOR BODY N LET 018011
C                LO = LLOC(NB+N) = LOCATION OF FIRST ELASTIC COORD. 018012
C                        IN STATE VECTOR 018013
C                LE = LEND(NB+N) = NUMBER OF ELASTIC COORD. IN Y 018014
C      (GGS(J,1,N),J=1,LE) = ROW J OF COF11(1,1,N)*Y(LO) 018015
C      (GGS(J,2,N),J=1,LE) = ROW J OF COF22(1,1,N)*Y(LO) 018016
C      (GGS(J,3,N),J=1,LE) = ROW J OF COF33(1,1,N)*Y(LO) 018017
C      (GGS(J,4,N),J=1,LE) = ROW J OF COF12(1,1,N)*Y(LO) 018018
C      (GGS(J,5,N),J=1,LE) = ROW J OF COF13(1,1,N)*Y(LO) 018019
C      (GGS(J,6,N),J=1,LE) = ROW J OF COF23(1,1,N)*Y(LO) 018020
C      (GGS(J,7,N),J=1,LE) = CCL J OF Y(LO)**T*COF12(1,1,N) 018021
C      (GGS(J,8,N),J=1,LE) = CCL J OF Y(LO)**T*COF13(1,1,N) 018022
C      (GGS(J,9,N),J=1,LE) = CCL J OF Y(LO)**T*COF23(1,1,N) 018023
C 018024
C 018025

```

C	KINEMATIC DATA INPUTTED AND COMPUTED	017826
C	-----	017827
C		017828
C	BETAH = CONTIGUOUS HINGE FRAME ORIENTATION DATA ARRAY, REAL*8	017829
C	PARAMETERS STORED IN /SPECIF/, DIMENSIONED (6,NHMAX)	017830
C	INITIAL VALUES INPUTTED IN DYNISJ	017831
C	INSTANTANEOUS VALUES ARE STRIPPED FROM THE STATE VECTOR	017832
C	ARRAY Y IN KUTDH	017833
C	(BETAH(I,N),I=1,3) = 3 SUCCESSIVE EULER ANGLES REQUIRED	017834
C	TO ORIENT Q-FRAME RELATIVE TO	017835
C	P-FRAME AT HINGE POINT N	017836
C	(BETAH(I,N),I=4,6) = 3 POSITION COORDINATES OF THE ORIGIN	017837
C	OF THE Q-FRAME RELATIVE TO THE	017838
C	P-FRAME AT HINGE POINT N	017839
C	BETAHD = CONTIGUOUS HINGE FRAME RELATIVE RATE DATA ARRAY, REAL*8	017840
C	PARAMETERS STORED IN /SPECIF/, DIMENSIONED (6,NHMAX)	017841
C	INITIAL VALUES INPUTTED IN DYNISJ	017842
C	INSTANTANEOUS VALUES ARE STRIPPED FROM THE STATE VECTOR	017843
C	DOT ARRAY YDT IN YDOT	017844
C	BETAHD(I,N) = FIRST TIME DERIVATIVE OF BETAH(I,N)	017845
C	((I=1,6),N=1,NH)	017846
C	AM0 = MOMENTUM WHEEL DATA ARRAY, REAL*3 PARAMETERS STORED IN	017847
C	/SPECIF/, DIMENSIONED (2,NM*MAX), INPUT DATA FROM DYNISJ	017848
C	THESE PARAMETERS ARE NOT FUNCTIONS OF TIME	017849
C	AM0(1,M) = INITIAL SPIN RATE FOR WHEEL M	017850
C	AM0(2,M) = SPIN INERTIA FOR WHEEL M	017851
C		017852
C		017853
C	RUTDH = TRANSFORMATION MATRICES COMPUTED THEREIN	017854
C	-----	017855
C	THE FOLLOWING ARRAYS ARE FUNCTIONS OF TIME. ELEMENTS OF THE STATE	017856
C	VECTOR Y ARE USED IN KUTDH TO COMPUTE THE KINEMATIC	017857
C	TRANSFORMATION MATRICES ASSOCIATED WITH HINGE POINT AND BODY	017858
C	FIXED REFERENCE FRAMES, SET DEBUG(44) = .TRUE. TO PRINT ARRAYS	017859
C		017860
C	BETAH = EULER ANGLES UPDATED FROM STATE VECTOR Y AT TIME T IN	017861
C	SUBROUTINE KUTDH	017862
C	CH = REFERENCE POINT, HINGE POINT POSITION VECTOR ARRAY	017863
C	DIMENSIONED (3,7*(NHMAX-1)), STORED IN /SPECIF/. ACROSS	017864
C	HINGE POINT NOH, NOH=2,3,...,NH THE FOLLOWING POSITION	017865
C	VECTORS ARE COMPUTED. LE1	017866
C	LE1 = 7*(NOH-2) + 1	017867
C	THEN	017868
C	(CH(J,LE1+0),J=1,3) = 3 COORDINATES OF THE ORIGIN OF THE	017869
C	Q-FRAME ON BODY N=ITOPOL(1,NOH) WRT THE	017870
C	BODY N REFERENCE IN THE UNDEFORMED	017871
C	STATE (MSMODL OR MSMODC)	017872
C	(CH(J,LE1+1),J=1,3) = 3 COORDINATES OF THE ORIGIN OF THE	017873
C	P-FRAME ON BODY M=ITOPOL(2,NOH) WRT THE	017874
C	BODY M REFERENCE IN THE UNDEFORMED	017875
C	STATE (MSMODL OR MSMODC)	017876
C	(CH(J,LE1+2),J=1,3) = 3 COORDINATES OF THE ORIGIN OF THE	017877
C	Q-FRAME ON BODY N=ITOPOL(1,NOH) WRT THE	017878
C	BODY N REFERENCE, DEFORMED STATE	017879
C	(CH(J,LE1+3),J=1,3) = 3 COORDINATES OF THE ORIGIN OF THE	017880
C	P-FRAME ON BODY M=ITOPOL(2,NOH) WRT THE	017881
C	BODY M REFERENCE, DEFORMED STATE	017882
C	(CH(J,LE1+4),J=1,3) = 3 COORDINATES OF THE ORIGIN OF THE	017883
C	Q-FRAME WRT THE P-FRAME AT HINGE	017884
C	POINT NOH	017885

C	(OH(J,LD1+5),J=1,3) = 3 COORDINATES OF THE ORIGIN OF THE	017886
C	BODY N=ITOPOL(1,NOH) FIXED REFERENCE	017887
C	WRT THE BODY M=ITOPOL(2,NOH) FIXED	017888
C	REFERENCE	017889
C	(OH(J,LD1+6),J=1,3) = 3 COORDINATES OF THE POSITION VECTOR	017890
C	FROM THE ORIGIN OF THE BODY	017891
C	N=ITOPOL(1,NOH) FIXED REFERENCE TO THE	017892
C	ORIGIN OF THE BODY M=ITOPOL(2,NOH)	017893
C	FIXED REFERENCE WRT THE INERTIAL	017894
C	REFERENCE FRAME	017895
C	RH = BODY FIXED REFERENCE FRAME, HINGE POINT REFERENCE FRAME	017896
C	TRANSFORMATION MATRICES, DIMENSIONED (3,3,6*(NHMAX-1)),	017897
C	STORED IN /SPECIF/. ACROSS HINGE POINT NOH, NOH=2,...,NH	017898
C	THE FOLLOWING COORDINATE TRANSFORMATION MATRICES ARE	017899
C	COMPUTED. LET	017900
C	LR1 = 6*(NOH-2) + 1	017901
C	THEN	017902
C	((RH(I,J,LR1+0),I=1,3),J=1,3) = 9 ELEMENTS OF TRANSFORMATION	017903
C	MATRIX USED TO TAKE VECTORS FROM	017904
C	Q-FRAME COORDINATES AT HINGE POINT NOH	017905
C	TO BODY N=ITOPOL(1,NOH) FIXED REFERENCE	017906
C	FRAME COORDINATES IN BODY N UNDEFORMED	017907
C	STATE	017908
C	((RH(I,J,LR1+1),I=1,3),J=1,3) = 9 ELEMENTS OF TRANSFORMATION	017909
C	MATRIX USED TO TAKE VECTORS FROM	017910
C	P-FRAME COORDINATES AT HINGE POINT NOH	017911
C	TO BODY M=ITOPOL(2,NOH) FIXED REFERENCE	017912
C	FRAME COORDINATES IN BODY M UNDEFORMED	017913
C	STATE	017914
C	((RH(I,J,LR1+2),I=1,3),J=1,3) = TRANSFORMATION MATRIX Q-FRAME	017915
C	COORDINATES AT HINGE POINT NOH TO	017916
C	BODY N=ITOPOL(1,NOH) FIXED COORDINATES	017917
C	IN BODY N DEFORMED STATE	017918
C	((RH(I,J,LR1+3),I=1,3),J=1,3) = TRANSFORMATION MATRIX P-FRAME	017919
C	COORDINATES AT HINGE POINT NOH TO	017920
C	BODY M=ITOPOL(2,NOH) FIXED COORDINATES	017921
C	IN BODY M DEFORMED STATE	017922
C	((RH(I,J,LR1+4),I=1,3),J=1,3) = TRANSFORMATION MATRIX Q-FRAME	017923
C	COORDINATES TO P-FRAME COORDINATES AT	017924
C	HINGE POINT NOH	017925
C	((RH(I,J,LR1+5),I=1,3),J=1,3) = TRANSFORMATION MATRIX FROM	017926
C	BODY N=ITOPOL(1,NOH) FIXED COORDINATES	017927
C	TO BODY M=ITOPOL(2,NOH) FIXED	017928
C	COORDINATES IN THE DEFORMED STATE OF	017929
C	BODIES N AND M	017930
C	ROL = BODY FIXED TO INERTIAL FRAME TRANSFORMATION MATRIX ARRAYS	017931
C	DIMENSIONED (3,3,NBMAX), STORED IN /BHBSRO/	017932
C	((ROL(I,J,N),I=1,3),J=1,3) = 9 ELEMENTS OF THE TRANSFORMATION	017933
C	MATRIX USED TO TRANSFORM VECTORS	017934
C	FROM BODY N FIXED REFERENCE FRAME	017935
C	COORDINATES TO THE INERTIAL FIXED	017936
C	REFERENCE FRAME	017937
C	COL = POSITION VECTOR ARRAY, INERTIAL ORIGIN TO BODY FIXED	017938
C	FRAME ORIGIN, DIMENSIONED (3,NBMAX), STORED IN /BHBSRO/	017939
C	(COL(I,N),I=1,3) = 3 COMPONENTS OF THE POSITION	017940
C	VECTOR FROM THE INERTIAL ORIGIN	017941
C	TO THE ORIGIN OF THE BODY N FIXED	017942
C	REFERENCE WRT THE INERTIALLY	017943
C	FIXED REFERENCE	017944
C	FIN = PI INVERSE TRANSFORMATIONS FOR HINGE POINT KINEMATICS	017945

```

C          DIMENSIONED (3,3,NHMAX), STORED IN /PINRP/,      017946
C          SEE APPENDIX B VOL 1                               017947
C          ((PIN(1,J,NOH),I=1,3),J=1,3) = PI INVERSE TRANSFORMATION USED 017948
C          TO RELATE THE EULER RATES ABOUT THE                017949
C          SKEWED AXES AT HINGE POINT NOH TO THE              017950
C          COMPONENTS OF THE RELATIVE ANGULAR                  017951
C          VELOCITY VECTOR OF THE Q-FRAME WRT THE              017952
C          P-FRAME AT HINGE POINT NOH.                          017953
C          RP2 = ARRAYS USED TO STORE MATRICES REQUIRED FOR THE 017954
C          RP3   COMPUTATION OF THE TIME DERIVATIVE OF THE PI INVERSE 017955
C          MATRIX. BOTH DIMENSIONED (3,3,NHMAX) AND STORED    017956
C          IN /PINRP/                                          017957
C          ((RP2(1,J,NOH),I=1,3),J=1,3) = 3 BY 3 COEFFICIENT MATRIX FOR 017958
C          THETA SUB 2 OUT IN EQ C-16 VOL 1 USED              017959
C          TO GET PI INVERSE OUT AT HINGE NCH                  017960
C          ((RP3(1,J,NOH),I=1,3),J=1,3) = 3 BY 3 COEFFICIENT MATRIX FOR 017961
C          THETA SUB 3 OUT IN EQ C-16 VOL 1 USED              017962
C          TO GET PI INVERSE OUT AT HINGE NCH                  017963
C                                                                017964
C                                                                017965
C          FCTOS - SENSOR FRAME LOCATION AND ORIENTATION      017966
C          -----                                             017967
C          RS = TRANSFORMATION MATRICES FROM SENSOR POINT TO BODY FIXED 017968
C          REFERENCE FRAME, DIMENSIONED (3,3,2*NSPMAX), STORED 017969
C          IN /SPECIF/. SET DEBUG(46) = .TRUE. TO PRINT        017970
C          OS = COORDINATES OF POSITION VECTOR FROM BODY FIXED REFERENCE 017971
C          TO SENSOR POINT WRT TO BODY FRAME, STORED IN /SPECIF/ 017972
C          DIMENSIONED (3,2*NSPMAX). SET DEBUG(46) = .TRUE. TO PRINT 017973
C          FOR SENSOR POINT NS. NS=1,2,....,NSPT LET          017974
C          N = IFTSMX(NS)                                       017975
C          LR1 = 2*NS-1                                         017976
C          LR2 = 2*NS                                           017977
C          ((RS(1,J,LR1),I=1,3),J=1,3) = TRANSFORMATION MATRIX WHICH 017978
C          TAKES VECTORS FROM SENSOR POINT NS FIXED            017979
C          COORDINATES TO BODY N FIXED COORDINATES             017980
C          IN THE UNDEFORMED STATE OF BODY N                   017981
C          ((RS(1,J,LR2),I=1,3),J=1,3) = TRANSFORMATION MATRIX SENSOR 017982
C          POINT NS FRAME TO BODY N FRAME IN DEFORMED          017983
C          STATE OF BODY N                                       017984
C          (OS(1,LR1),I=1,3) = POSITION VECTOR FROM BODY N      017985
C          REFERENCE POINT TO SENSOR POINT NS WRT THE          017986
C          BODY N FIXED REFERENCE IN UNDEFORMED STATE          017987
C          (OS(1,LR2),I=1,3) = POSITION VECTOR FROM BODY N      017988
C          REFERENCE POINT TO SENSOR POINT MS WRT THE          017989
C          BODY N FIXED REFERENCE IN DEFORMED STATE            017990
C                                                                017991
C                                                                017992
C          EHGENR - HINGE POINT KINEMATICS COEFFICIENTS      017993
C          -----                                             017994
C                                                                017995
C          BH = VELOCITY TRANSFORMATION ARRAY. THE NON-ZERO PARTITIONS 017996
C          OF THE VELOCITY TRANSFORMATION EQUATION II-84 ARE STORED 017997
C          IN THIS ARRAY. THE ARRAY IS STORED IN /USBRD/ AND    017998
C          DIMENSIONED (6,6+NMOROD,2*NHMAX-1).                017999
C          SET DEBUG(45) = .TRUE. TO PRINT                     018000
C          ((BH(1,J,1),I=1,6),J=1,6+IRGFLX(1)) = COMPONENTS OF THE (1,1) 018001
C          PARTITION OF THE COEFFICIENT MATRIX IN              018002
C          EQUATION II-84. RELATES TO THE Q-FRAME IN           018003
C          BODY 1 AT HINGE POINT 1, EVALUATED BY USE           018004
C          OF EQUATION II-83                                     018005

```



```

C          AT HINGE POINT NOH, NOH=2,3,...,NH LET                                C18006
C          LQ = 2*NOH-2                                                            C18007
C          LP = 2*NOH-1                                                            C18008
C          N = ITOPCL(1,NOH) (Q-FRAME IN BODY N)                                C18009
C          M = ITOPCL(2,NOH) (P-FRAME IN BODY M)                                C18010
C          NMQ = IROFLX(N) (NMQ MODES FOR BODY N)                               C18011
C          NMP = IROFLX(M) (NMP MODES FOR BODY M)                               C18012
C          ((QH(1,J,LP),I=1,6),J=1,6+NMP) = COMPONENTS OF THE (NOH,M)           C18013
C          PARTITION OF THE COEFFICIENT MATRIX IN                               C18014
C          EQUATION 11-84. RELATES TO THE P-FRAME IN                             C18015
C          BODY M AT HINGE POINT NOH, EVALUATED BY                               C18016
C          USE OF EQUATION 11-82                                                    C18017
C          ((QH(1,J,LQ),I=1,6),J=1,6+NMQ) = COMPONENTS OF THE (NOH,N)           C18018
C          PARTITION OF THE COEFFICIENT MATRIX IN                               C18019
C          EQUATION 11-82. RELATES TO THE Q-FRAME IN                             C18020
C          BODY N AT HINGE POINT NOH, EVALUATED BY                               C18021
C          USE OF EQUATION 11-82                                                    C18022
C                                                                                   C18023
C                                                                                   C18024
C          ESSENK = SENSOR POINT KINEMATICAL COEFFICIENTS                        C18025
C          -----                                                                C18026
C          ES = VELOCITY TRANSFORMATION MATRIX FOR SENSOR POINT FRAMES            C18027
C          DIMENSIONED (6,6+NMQ+600,NSPT) STORED IN /UHBSRD/                     C18028
C          SET DEBUG(47) = .TRUE. TO PRINT                                         C18029
C          FOR SENSOR POINT NS, NS=1,2,...,NSPT                                   C18030
C          N = IFISMW(NS) SENSOR POINT NS IN BODY N                               C18031
C          LE = IROFLX(N) LE MODES FOR BODY N                                     C18032
C          ((ES(1,J,NS),I=1,6),J=1,6+LE) = TRANSFORMATION MATRIX USED TO        C18033
C          OBTAIN THE COMPONENTS OF THE BODY N                                     C18034
C          INERTIAL ANGULAR AND LINEAR VELOCITY                                   C18035
C          VECTORS IN SENSOR NS FRAME COORDINATES                                C18036
C          SEE PAGE 17, VOL 11                                                    C18037
C                                                                                   C18038
C          USER DEFINE CONTROL SYSTEM DATA                                       C18039
C          -----                                                                C18040
C          CNTDTA = REAL*8 ARRAY OF CONSTANTS READ IN, IN DYNSTD AND STORED        C18041
C          IN /CONPAR/. THESE ARE USER DEFINED PARAMETERS.                       C18042
C          DIMENSIONED (MAXCNT)                                                    C18043
C          DISCOS ASSUMES THAT:                                                    C18044
C          CNTDTA(1),I=1,NDELTA = INITIAL CONDITIONS FOR THE                     C18045
C          NDELTA DIFFERENTIAL EQUATIONS                                           C18046
C          CODED BY THE USER IN CNTRL                                             C18047
C          CNTDTA(1),I.GT.NDELTA = CONSTANTS THAT THE USER WILL                  C18048
C          NEED FOR THE USER CODED                                                 C18049
C          EQUATIONS                                                                C18050
C                                                                                   C18051
C                                                                                   C18052
C          STATE VECTOR AND ITS FIRST TIME DERIVATIVE                           C18053
C          -----                                                                C18054
C          THE REAL*8 ARRAYS Y AND YDT ARE STORED IN /VECTOR/ AND                  C18055
C          DIMENSIONED (KY). IN SEVERAL ROUTINES THE SYMBOLIC NAME YD             C18056
C          IS USED INSTEAD OF YDT. SET DEBUG(39)=.TRUE. TO PRINT                   C18057
C          INSTANTANEOUS VALUES OF ELEMENTS IN Y AND YDT ARRAYS                  C18058
C                                                                                   C18059
C          (Y(I),I=LOCU(N),LOCU(N)+2) = COMPONENTS OF INERTIAL ANGULAR            C18060
C          VELOCITY VECTOR OF BODY N WRT THE                                       C18061
C          BODY N REFERENCE                                                         C18062
C          (Y(I),I=LOCU(N)+3,LOCU(N)+5) = COMPONENTS OF INERTIAL LINEAR           C18063
C                                                                                   C18064
C                                                                                   C18065

```

```

C                                     VELOCITY VECTOR OF BODY N WRT THE      018066
C                                     BODY N REFERENCE                        018067
C (Y(I),I=LCCU(N)+6,LCCU(N)+6 = GENERALIZED MODAL VELOCITIES           018068
C                                     -1+IRGFLX(N)) ASSOCIATED WITH THE FLEXURAL MODES 018069
C                                     OF BODY N. (IN ASCENDING ORDER AS      018070
C                                     READ IN, NONE IF BODY N RIGID)         018071
C (Y(I),I=LCCU(N)+6+IRGFLX(N),= SPIN RATES OF THE ACTIVE WHEELS        018072
C                                     LCCU(N)+6+IRGFLX(N) IN BODY N. (IN ASCENDING ORDER AS 018073
C                                     -1+NMUW(2,N)) READ IN, NONE IF NMUW(2,N)=0) 018074
C (Y(I),I=LCCU(NB+N), = GENERALIZED MODAL DISPLACEMENTS               018075
C                                     LCCU(NB+N)-1+IRGFLX(N)) ASSOCIATED WITH THE FLEXURAL MODES 018076
C                                     OF BODY N. (IN ASCENDING ORDER AS      018077
C                                     READ IN, NONE IF BODY N RIGID)         018078
C (Y(I),I=LCCU(2*NB+1), = BETA COORDINATES USED TO DEFINE              018079
C                                     LCCU(2*NB+1)-1+NBETA) UNCONSTRAINED AND RHEOMICALLY 018080
C                                     CONSTRAINED MOTION BETWEEN BODIES      018081
C                                     WHICH ARE CONTIGUOUS.                  018082
C                                     LET:                                   018083
C                                     NUM=BETA COORDINATE NUMBER FOR          018084
C                                     DEGREE OF FREEDOM I-1 AT               018085
C                                     HINGE POINT N. THE                     018086
C                                     NUMBERING ALGORITHM IS                 018087
C                                     NUM = 0                               018088
C                                     DO 1 N=1,NH                           018089
C                                     DO 1 I=2,7                             018090
C                                     IF (IHDATA(1,N).EQ.1) GO TO 1          018091
C                                     NUM = NUM+1                          018092
C                                     1 CONTINUE                             018093
C (Y(I),I=LCCU(2*NB+2), = STATE VARIABLE COORDINATES WHICH              018094
C                                     LCCU(2*NB+2)-1+NDELTA) ARE ASSOCIATED WITH THE NDELTA FIRST 018095
C                                     ORDER DIFFENTIAL EQUATIONS CODED BY 018096
C                                     THE USER IN CONTRL                   018097
C (YC(I),I=1,NEG) = FIRST TIME DERIVATIVE OF THE                       018098
C                                     COORDINATE Y(I) IN THE STATE VECTOR 018099
C                                     018100
C                                     018101
C FINDU - FIND SET OF INDEPENDENT COORDINATES IN STATE VECTOR          018102
C -----                                                                018103
C                                     018104
C IFLAG = DUMMY VARIABLE SET IN CALLING ROUTINE                        018105
C IFLAG = 0 INITIAL PASS THROUGH COMPUTE MAGNITUDES OF                 018106
C INDEPENDENT COORDINATES OF STATE VECTOR                             018107
C IFLAG = 1 NOT AN INTERMEDIATE STEP THROUGH RUNGE-KUTTA             018108
C INTEGRATION CYCLE, GET THE DEPENDENT COORDINATE                     018109
C MAGNITUDES AND PUT THEM IN THE STATE VECTOR.                       018110
C ALSO CHECK TO SEE IF OPTIMUM SET OF INDEPENDENT                     018111
C COORDINATES BEING USED                                              018112
C IFLAG = 2 AN INTERMEDIATE STEP THROUGH RUNGE-KUTTA; GET            018113
C DEPENDENT COORDINATES, USE SAME SET OF                              018114
C INDEPENDENT COORDINATES AS COMPUTED ON LAST                         018115
C PASS THROUGH WITH IFLAG=1                                           018116
C BW = MULTI-PURPOSE ARRAY STORED IN /AMJBW/                          018117
C DIMENSIONED (C*NHMAX,NEMAX*(6+NMU300)+NMWMAX).                     018118
C AT THE DEBUG#36 PRINT POINT IN FINDU LABELED                        018119
C 'THE BW MATRIX IS'                                                  018120
C ((BW(1,J),I=1,6*NH),J=1,NC) = THE VELOCITY TRANSFORMATION        018121
C MATRIX AS DEFINED BY EQUATION II-24                                018122
C IN VCL 1                                                            018123
C INDEP = INDEPENDENT/DEPENDENT COORDINATE TABLE, AN INTEGER*4     018124
C ARRAY STORED IN /VINDEP/ AND DIMENSIONED (KY). USED FOR           018125

```

```

C      THE CASES WHEN                                C18126
C      IFLAG = 1 OR 2                                C18127
C      TO DEFINE WHICH COORDINATES IN THE STATE VECTOR Y ARE C18128
C      TO BE TREATED AS BEING INDEPENDENT AND WHICH DEPENDENT C18129
C      INDEP(LU) = 0 IMPLIES THE COORDINATE Y(LU) IS FOR THE C18130
C      PRESENT INTEGRATION STEP A DEPENDENT C18131
C      COORDINATE C18132
C      INDEP(LU) = 1 IMPLIES THE COORDINATE Y(LU) IS FOR THE C18133
C      PRESENT INTEGRATION STEP AN INDEPENDENT C18134
C      COORDINATE LU=1,...,NU C18135
C      AT THE END OF EACH INTEGRATION STEP A CHECK IS MADE IN C18136
C      FINDU TO ASCERTAIN IF A BETTER SET OF INDEPENDENT C18137
C      COORDINATES CAN BE FOUND; IF SO, ARRAY INDEP IS UPDATED C18138
C      THE ALGORITHM FOR COMPUTING THE SET OF INDEPENDENT COORDS. C18139
C      HAS A BUILT IN BIAS. THE DEPENDENT COORDS. USED TO C18140
C      ACCOUNT FOR CONSTRAINTS AT HINGE POINT 1 ARE CHOSEN C18141
C      FROM THE VELOCITY COMPONENTS USED TO DEFINE THE INERTIAL C18142
C      VELOCITY OF THE Q FRAME RELATIVE TO THE P FRAME AT C18143
C      HINGE POINT 1. IF THIS CAN'T BE DONE ANY COORDINATE THEN C18144
C      BECOMES FAIR GAME FOR USE. C18145
C      C18146
C      C18147
C      YDOT - FIRST TIME DERIVATIVE OF STATE VECTOR C18148
C      ----- C18149
C      SUBROUTINE YDOT DIRECTS THE COMPUTATION REQUIRED FOR THE C18150
C      DETERMINATION OF THE YDT ARRAY. C18151
C      GGV = TORQUE ARRAY, DIMENSIONED (NBMAX*(J+NMDBOD)+NMWMAX) C18152
C      THIS ARRAY IS NOT STORED IN COMMON. IN YDOT IT IS C18153
C      INITIALIZED TO ZERO ON EACH PASS THROUGH. IN SUBROUTINE C18154
C      GRVGRD THE GENERALIZED FORCE ASSOCIATED WITH GRAVITY C18155
C      GRADIENT EFFECTS ACTING ON EACH STATE VECTOR C18156
C      COORDINATE IS COMPUTED AND STORED IN GGV. IN SUBROUTINE C18157
C      TORQUE ALL OTHER SYSTEM FORCE AND TORQUES ARE COMPUTED C18158
C      AND ADDED ON TO GGV, DEBUG(50 AND 57) = .TRUE. TO PRINT C18159
C      (GGV(J),J=1,NU) = TOTAL GENERALIZED FORCE ASSOCIATED WITH THE C18160
C      J-TH STATE VECTOR COORDINATE EQUATION C18161
C      BMB = B**(-1)*B**T MATRIX IN EQUATION 11-6, VOL 1, COMPUTED C18162
C      IN GETBMB AND PRINTED IF DEBUG(48)=.TRUE., STORED C18163
C      TEMPORARILY IN THE BW ARRAY VIA AN EQUIVALENCE C18164
C      BM = B**(-1) MATRIX USED ALSO IN EQUATION 11-6, COMPUTED C18165
C      IN GETBMB AND STORED TEMPORARILY IN ANOTHER PART OF C18166
C      THE BW ARRAY C18167
C      BDTF = FIRST DERIVATIVE OF THE HINGE POINT KINEMATIC COEFFICIENT C18168
C      MATRIX ASSOCIATED WITH THE P-FRAME, 3*3T(EQ. C-1, VOL 1) C18169
C      DIMENSIONED (3,3+NMDBOD), STORED LOCALLY C18170
C      BDTF FOR EACH HINGE POINT PRINTED IF DEBUG(51)=.TRUE. C18171
C      BDTQ = FIRST DERIVATIVE OF THE HINGE POINT KINEMATIC COEFFICIENT C18172
C      MATRIX ASSOCIATED WITH THE Q-FRAME, 3*3T(EQ. C-2, VOL 1) C18173
C      DIMENSIONED (3,3+NMDBOD), STORED LOCALLY C18174
C      BDTQ FOR EACH HINGE POINT PRINTED IF DEBUG(51)=.TRUE. C18175
C      ALAM = LAGRANGE MULTIPLIERS, FORCES AND TORQUES OF CONSTRAINT C18176
C      STORED IN A COMPACT ARRAY IN /LAMJUA/ WHICH IS C18177
C      DIMENSIONED (6*NHMAX), DEBUG(52) = .TRUE. TO PRINT C18178
C      LET IC = IV(1,L) THEN C18179
C      ALAM(IC) = CONSTRAINT FORCE OR TORQUE ACTING TO RESTRICT C18180
C      MOTION AT HINGE POINT L. NOTE THAT C18181
C      THE LAGRANGE MULTIPLIERS COMPUTED IN BAKSLV ARE C18182
C      THE CONSTRAINT FORCES AND TORQUES C18183
C      V = ALAM VIA AN EQUIVALENCE IN YDOT C18184
C      IV = LAGRANGE MULTIPLIER NUMBERING TABLE, INTEGER ARRAY STORED C18185

```

C	IN /ZVCNS/ DIMENSIONED (6,NHMAX) TO PROVIDE A COMPACT	C18186
C	NUMBERING SEQUENCE FOR THE LAGRANGE MULTIPLIERS	C18187
C	IV(I,L) = NUMBER ASSIGNED INTERNALLY TO THE LAGRANGE	C18188
C	MULTIPLIER ASSOCIATED WITH DEGREE OF FREEDOM I	C18189
C	AT HINGE POINT L, EQUALS ZERO IF DEGREE OF	C18190
C	FREEDOM I AT HINGE POINT L UNCONSTRAINED.	C18191
C	COMPUTED IN GETHMD	C18192
C		C18193
C		C18194
C	USER SUPPLIED SUBROUTINES	C18195
C	-----	C18196
C		C18197
C	RATHER THAN ATTEMPT THE IMPOSSIBLE TASK OF FORESEEING ALL POSSIBLE	C18198
C	SIMULATION NEEDS OF ALL POTENTIAL USERS	C18199
C		C18200
C	'DISCOS DEMANDS USER INTERFACE'	C18201
C		C18202
C	THE USER MUST CODE IN FORTRAN ALL EQUATIONS REQUIRED TO DESCRIBE	C18203
C	ALL NON-GYROSCOPIC AND NON-GRAVITY GRADIENT EFFECTS REQUIRED	C18204
C	FOR SIMULATION PURPOSES	C18205
C		C18206
C	DISCOS ATTEMPTS TO PROVIDE THE USER WITH A CLEAN INTERFACE BETWEEN	C18207
C	PARAMETERS WHICH WOULD NORMALLY BE CONSIDERED TO BE PHYSICALLY	C18208
C	REALIZABLE AND THE GENERALIZED FORCE AND TORQUE PARAMETERS	C18209
C	USED FOR ACTUAL COMPUTATION. THIS IS DONE VIA A SET OF	C18210
C	FUNCTIONS AND SUBROUTINES WHICH REQUIRE THE USER TO DEFINE	C18211
C	CERTAIN PHYSICALLY REALIZABLE PARAMETERS; THESE PARAMETERS	C18212
C	ARE THEN AUTOMATICALLY TRANSFORMED INTO THE APPROPRIATE	C18213
C	GENERALIZED PARAMETERS REQUIRED FOR COMPUTATION.	C18214
C		C18215
C	ALL USER REQUIRED INPUT DATA IS NORMALLY READ FROM CARDS BY	C18216
C	FORMATTED READ STATEMENTS IN SUBROUTINE DYN510. IT IS HERE	C18217
C	THAT THE USER DEFINES	C18218
C	1) NDELTA - TOTAL NUMBER OF ADDITIONAL FIRST ORDER	C18219
C	DIFFERENTIAL EQUATIONS WHICH WILL BE USER	C18220
C	DEFINED IN SUBROUTINE CNTRL	C18221
C	2) CNTOTA(I),I,LE,NDELTA - INITIAL CONDITIONS FOR THE USER	C18222
C	DEFINED DIFFERENTIAL EQUATIONS	C18223
C	3) CNTOTA(I),I,GT,NDELTA - ALL OTHER PARAMETRIC MAGNITUDES	C18224
C	OF PARAMETERS USED IN THE USER	C18225
C	DEFINED EQUATIONS	C18226
C		C18227
C	ALL USER REQUIRED STATE VARIABLE DATA CAN BE OBTAINED BY SIMPLY	C18228
C	INCLUDING THE APPROPRIATE COMMON BLOCKS IN THE SUBROUTINES	C18229
C	BEING USER MODIFIED. DATA MAY BE TRANSFERRED BETWEEN THE	C18230
C	USER MODIFIABLE ROUTINES BY EITHER USING AVAILABLE SPACE	C18231
C	IN /CONPAR/ OR BY DEFINING NEW SPECIAL PURPOSE COMMON BLOCKS	C18232
C		C18233
C		C18234
C	SUBROUTINE CNTRL USER DEFINED DIFFERENTIAL EQUATIONS	C18235
C	-----	C18236
C		C18237
C	USER SPECIFICATION OF ALL NON-GYROSCOPIC AND NON-GRAVITY GRADIENT	C18238
C	FORCES AND TORQUES FREQUENTLY REQUIRES THE CAPABILITY TO ADD	C18239
C	AND SIMULTANEOUSLY SOLVE SEVERAL ADDITIONAL DIFFERENTIAL	C18240
C	EQUATIONS. THESE ARE REFERRED TO AS 'CONTROL EQUATIONS' SINCE	C18241
C	IN MOST SPACECRAFT SIMULATION PROBLEMS THEY ARE THE EQUATIONS	C18242
C	WHICH SIMULATE THE ACTIVE ON-BOARD CONTROL SYSTEM. THE TOTAL	C18243
C	NUMBER OF USER DEFINED FIRST ORDER DIFFERENTIAL EQUATIONS	C18244
C	IS 'NDELTA'. THIS PARAMETER IS SET VIA DATA INPUT IN DYN510	C18245

```

C      *** IF TRANSFER FUNCTIONS ARE TO BE GENERATED THE USER MUST ADHERE 018246
C      TO THE RULE THAT ALL CONTROL INFORMATION (SYSTEM STATE DATA) 018247
C      COMES THROUGH SENSOR SIGNALS. IF THIS IS NOT DONE THE 018248
C      NUMERICAL LINEARIZATION ALGORITHM CANNOT DIFFERENTIATE 018249
C      BETWEEN CONTROL TORQUES AND GYROSCOPIC TORQUES, AND HENCE 018250
C      CANNOT OPEN FEEDBACK LOOPS WHEN OPEN AND QUASI-OPEN LOOP 018251
C      TRANSFER FUNCTIONS ARE CALLED FOR 018252
C 018253
C      IN ADDITION TO DEFINING ALL CONTROL EQUATIONS IN CONTRL INTERNAL 018254
C      COMPUTATION LOGIC REQUIRES THAT THE USER ALSO DEFINE IN CONTRL 018255
C      NXSS - TOTAL NUMBER OF SENSOR SIGNALS TO BE DEFINED 018256
C      IN EQADD (DEFAULT NXSS=1) 018257
C      NBTQ - TOTAL NUMBER OF TORQUE SIGNALS TO BE DEFINED 018258
C      IN EQADD (DEFAULT NBTQ=0) 018259
C 018260
C      DISCOS ASSUMES THAT THE NDELTA STATE VARIABLES ASSOCIATED WITH THE 018261
C      NDELTA USER DEFINED CONTROL EQUATIONS ARE INDEPENDENT. TO 018262
C      SIMULTANEOUSLY SOLVE THESE EQUATIONS ALONG WITH THE DISCOS 018263
C      DERIVED PLANT EQUATIONS THE FOLLOWING STORAGE CONVENTION MUST 018264
C      BE ADHERED TO: 018265
C      LET 018266
C      LDEL = LLOC(2*NB+2)-1 (SEE DEFINITION ABOVE OF LLOC) 018267
C      THEN 018268
C      YOT(LDEL+1),I=1,NDELTA = USER DEFINED LINEAR OR NON-LINEAR 018269
C      FUNCTION OF TIME AND SYSTEM STATE. 018270
C      THIS IS THE I-TH USER DEFINED 018271
C      FIRST ORDER DIFFERENTIAL EQUATION 018272
C      WHICH WILL BE SIMULTANEOUSLY 018273
C      INTEGRATED WITH THE DISCOS DERIVED 018274
C      PLANT EQUATIONS 018275
C      Y(LDEL+1),I=1,NDELTA = LOCATION IN THE STATE VECTOR ARRAY 018276
C      OF THE STATE VARIABLE OBTAINED VIA 018277
C      NUMERICAL INTEGRATION OF THE I-TH 018278
C      USER DEFINED DIFFERENTIAL EQUATION 018279
C      AND 018280
C      CNTOTA(I),I=1,NDELTA = INITIAL CONDITION TO BE USED FOR 018281
C      THE I-TH USER DEFINED DIFFERENTIAL 018282
C      EQUATION. THESE PARAMETERS ARE SET 018283
C      VIA DATA INPUT IN DYN10 018284
C 018285
C      FREQUENTLY THE ONLY CONTROL SYSTEM DESCRIPTION AVAILABLE TO THE 018286
C      ANALYST IS A MULTILoop FEEDBACK BLOCK DIAGRAM CONSISTING 018287
C      OF TRANSFER FUNCTIONS, LOGIC GATES AND NON-LINEAR ELEMENTS. 018288
C      THE DISCOS USER MAY EITHER TRANSFORM THIS DESCRIPTION TO THE 018289
C      TIME DOMAIN AND DIRECTLY CODE THE RESULTANT DIFFERENTIAL 018290
C      EQUATIONS, OR SIMPLY INPUT THE VARIOUS TRANSFER FUNCTIONS AND 018291
C      LET THE COMPUTER TRANSFORM THEM AND SET UP THE APPROPRIATE 018292
C      DIFFERENTIAL EQUATIONS. THE USER MAY IF IT IS DESIRABLE CODE 018293
C      DIRECTLY SOME OF THE DIFFERENTIAL EQUATIONS AND LET THE 018294
C      COMPUTER GENERATE THE REST FROM INPUTTED TRANSFER FUNCTION 018295
C      DATA. OBVIOUSLY, THE USER MUST ADHERE TO CERTAIN PROGRAMMING 018296
C      RULES TO SET THIS CAPABILITY UP: 018297
C      1) THERE IS NO REAL LIMIT AS TO THE NUMBER OF TRANSFER 018298
C      FUNCTIONS TO BE INPUTTED OR TO THE DEGREE OF THEIR 018299
C      DENOMINATOR POLYNOMIALS. THIS IS USER SET IN CONTRL. EACH 018300
C      TRANSFER FUNCTION MUST BE GIVEN AS A POLYNOMIAL RATIO IN S 018301
C      OF THE FORM 018302
C 018303
C      
$$TF(S) = \frac{A(1) + A(2)*S + \dots + A(KS)*S^{*(KS-1)}}{\dots}$$
 018304
C 018305

```

```

C          B(1) + B(2)*S + ... + B(NS)*S**(NS-1)      018306
C                                                    018307
C          *HERE          (KS.LE.NS.AND.NS.GE.2)      018308
C      2) DEFINE THE FOLLOWING MAXIMUM SIZE PARAMETERS VIA DATA 018309
C          STATEMENTS IN SUBROUTINE CNTRL          018310
C          NPLY = TOTAL NUMBER OF TRANSFER FUNCTIONS TO BE INPUTTED 018311
C          I1ST = 0, FIRST PASS THROUGH CNTRL FLAG 018312
C          KRY = 1+THE DEGREE OF THE HIGHEST DEGREE DENOMINATOR 018313
C          POLYNOMIAL TO BE INPUTTED (USED TO SET DIMENSION 018314
C          OF ARRAY IN WHICH ALL POLY. COEFF. STORED) 018315
C          KCY = 2*NPLY 018316
C      3) USE THE FOLLOWING ACRONYMS TO DEFINE THE TRANSFER FUNCTION 018317
C          DATA STORAGE ARRAYS 018318
C          KPLY = TRANSFER FUNCTION DENOMINATOR DEGREE ARRAY, 018319
C          INPUTTED INTEGER*4 ARRAY DIMENSIONED (NPLY) 018320
C          KPLY(K) = ONE PLUS THE DEGREE OF THE DENOMINATOR 018321
C          OF THE K-TH INPUTTED TRANSFER 018322
C          FUNCTION, K=1,2,...,NPLY 018323
C          CPLY = TRANSFER FUNCTION POLYNOMIALS COEFFICIENTS ARRAY, 018324
C          INPUTTED REAL*8 ARRAY DIMENSIONED (KRY,KCY) 018325
C          CPLY(1,2*K-1) = COEFFICIENT OF THE S**(I-1) TERM 018326
C          IN THE DENOMINATOR OF THE K-TH 018327
C          INPUTTED TRANSFER FUNCTION 018328
C          CPLY(1,2*K) = COEFFICIENT OF THE S**(I-1) TERM 018329
C          IN THE NUMERATOR OF THE K-TH 018330
C          INPUTTED TRANSFER FUNCTION 018331
C          K=1,2,...,NPLY 018332
C          I=1,2,...,KPLY(K) 018333
C      4) INSERT THE FOLLOWING FORTRAN CODE IN CNTRL TO INPUT ALL 018334
C          TRANSFER FUNCTION DATA IN A DISCOS CONSISTANT FORMAT 018335
C          IF(I1ST.NE.0) GO TO 110 018336
C          I1ST = 1 018337
C          IF(NPLY.EQ.0) GO TO 110 018338
C          CALL ZERO(CPLY,KRY,KCY,KRY) 018339
C          DO 105 K=1,NPLY 018340
C          K2 = 2*K-1 018341
C          105 CALL READ(CPLY(1,K2),KPLY(K),N2,KRY,KCY) 018342
C          CALL WRITE(CPLY,KRY,KCY,4HCPLY,KRY) 018343
C          110 CONTINUE 018344
C      5) TO TRANSFORM S-DOMAIN TRANSFER FUNCTION DATA TO TIME DOMAIN 018345
C          CONTROL EQUATIONS USE 018346
C
C          SUBROUTINE TFPLY(A,B,U,X,NS,L) 018348
C
C          THE FOLLOWING BLOCK DIAGRAM SETUP IS ASSUMED BY TFPLY 018349
C
C          U(K,T) |----->| TF(K,S) |-----> X(K,T) 018350
C          |----->| TF(K,S) |-----> 018351
C          |----->| TF(K,S) |-----> 018352
C          |----->| TF(K,S) |-----> 018353
C          |----->| TF(K,S) |-----> 018354
C          |----->| TF(K,S) |-----> 018355
C          |----->| TF(K,S) |-----> 018356
C          |----->| TF(K,S) |-----> 018357
C
C          TFPLY MUST BE CALLED EXACTLY NPLY TIMES FROM CNTRL EVERY 018358
C          INTEGRATION STEP. ON THE K-TH CALL TO TFPLY FROM CNTRL 018359
C          TFPLY ASSUMES THAT THE FOLLOWING PARAMETERS HAVE BEEN PUT 018360
C          BY THE USER PROVIDED CODE INTO ITS DUMMY ARGUMENTS: 018361
C          A = COEFFICIENTS OF THE DENOMINATOR POLYNOMIAL OF 018362
C          TF(K,S); THAT IS COLUMN 2*K-1 OF ARRAY CPLY 018363
C          B = COEFFICIENTS OF THE NUMERATOR POLYNOMIAL OF 018364
C          TF(K,S); THAT IS COLUMN 2*K OF ARRAY CPLY 018365

```

```

C          U = INSTANTANEOUS VALUE OF THE USER DEFINED VARIABLE          018366
C          WHICH FEEDS INTO THE BLOCK DEFINED BY TRANSFER                018367
C          FUNCTION TF(K,S). U(K,T) MAY BE ANY LINEAR OR NON-            018368
C          LINEAR FUNCTION OF SYSTEM STATE VARIABLES DEFINED            018369
C          IN THE TIME DGMMAIN                                           018370
C          NS = 1+DEGREE OF THE DENOMINATOR POLYNOMIAL OF TF(K,S);      018371
C          THAT IS, KPLY(K)                                              018372
C          L = LOCATION IN THE STATE VECTOR ARRAY Y OF THE LEADING      018373
C          ELEMENT OF THE KPLY(K)-1 STATE VARIABLES TO BE               018374
C          ESTABLISHED BY TFPY FROM TF(K,S)                              018375
C          NOTE THAT THE NPLY CALLS TO TFPY WILL                         018376
C          ESTABLISH EXACTLY                                             018377
C          KPLY(1)*KPLY(2)+...+KPLY(NPLY)-NPLY                          018378
C          NEW CONTROL EQUATIONS AND ASSOCIATED                         018379
C          STATE VARIABLES                                              018380
C          WITH THE ABOVE DATA LOADED INTO THE DUMMY ARGUMENTS OF      018381
C          TFPY IT WILL DEFINE:                                          018382
C          YD(L-1+I),I=1,KPLY(K)-1 = INSTANTANEOUS VALUE OF THE        018383
C          FIRST TIME DERIVATIVE OF THE                                018384
C          I-TH STATE VARIABLE REQUIRED                                  018385
C          BY THE TRANSFORMATION OF                                     018386
C          S-DGMMAIN DATA TO THE TIME                                 018387
C          DGMMAIN                                                      018388
C          X = INSTANTANEOUS VALUE OF THE OUTPUT X(K,T) OF THE          018389
C          BLOCK DEFINED BY TRANSFER FUNCTION TF(K,S)                   018390
C          THE ALGORITHM USED TO GENERATE THE KPLY(K)-1 CONTROL          018391
C          EQUATIONS IS GIVEN AS FOLLOWS: (SEE PAGE 78, VOL I)          018392
C          AN=A(NS)                                                      018393
C          DO 10 I=1,NS                                                  018394
C          A(I)=A(I)/AN                                                  018395
C          10 B(I)=B(I)/AN                                              018396
C          BN=B(NS)                                                      018397
C          DO 20 I=2,NS                                                  018398
C          J = NS-I+1                                                    018399
C          K = L+I-2                                                      018400
C          IF(I.EQ.NS) GO TO 25                                          018401
C          20 YD(K) = -A(J)*Y(L) + Y(K+1) + (J(J)-A(J)*BN)*U          018402
C          25 YD(K) = -A(J)*Y(L) + (B(J)-A(J)*BN)*U                    018403
C          X = Y(L) + BN*U                                              018404
C          THE USER WILL HAVE TO GO THROUGH THIS ALGORITHM IF NON-ZERO 018405
C          INITIAL CONDITIONS FOR SELECTED CONTROL VARIABLES ARE        018406
C          REQUIRED TO AVOID PROLONGED TRANSIENT RESPONSE. AGAIN, ALL    018407
C          INITIAL CONDITIONS ARE DEFINED IN THE INPUT ARRAY CNIDTA     018408
C          018409
C          018410
C          FUNCTIONS ADT & ADUT DEFINE RHEONOMIC CONSTRAINTS           018411
C          -----                                                       018412
C          018413
C          ASSUME THAT AT HINGE POINT L, RELATIVE MOTION WITH RESPECT TO 018414
C          DEGREE OF FREEDOM I IS RHEONOMICALLY CONSTRAINED.           018415
C          LET                                                            018416
C          NN = 6*(L-1) + I                                              018417
C          THEN                                                            018418
C          ADUT(NN,T) = USER PRESCRIBED MAGNITUDE OF THE SECOND        018419
C          TIME DERIVATIVE OF BETAH(I,L) AT TIME T                     018420
C          ADT(NN,T) = USER PRESCRIBED MAGNITUDE OF THE FIRST          018421
C          TIME DERIVATIVE OF BETAH(I,L) AT TIME T                     018422
C          WHERE BETAH IS DEFINED ABOVE UNDER THE TITLE 'KINEMATIC DATA 018423
C          INPUTTED AND COMPUTED'                                         018424
C          018425

```

C IF ADT(NN,T) IS TO BE DEFINED BY A DISCONTINUOUS FUNCTION OF TIME 018426
 C IT IS RECOMMENDED THAT ADT(NN,T) BE OBTAINED VIA NUMERICAL 018427
 C INTEGRATION OF ADDT(NN,T) IN SUBROUTINE CONTRL. IF THIS 018428
 C IS NOT DONE NUMERICAL PROBLEMS CAN OCCUR. 018429
 C 018430
 C 018431
 C SUBROUTINE EXTOR EXTERNAL TORQUES AND FORCES 018432
 C ----- 018433
 C 018434
 C ALL EXTERNAL TORQUES AND FORCES ACTING ON THE SYSTEM MUST BE 018435
 C DEFINED IN SUBROUTINE EXTOR, SEE PAGE 12, VOL 11. EXTOR IS 018436
 C CALLED FROM SUBROUTINE TORQUE IMMEDIATELY AFTER RETURN FROM 018437
 C CONTRL. UPON ENTRY INTO EXTOR THE STATE VECTOR Y CONTAINS THE 018438
 C INSTANTANEOUS VALUES OF ALL PLANT AND CONTROL SYSTEM STATE 018439
 C VARIABLES. THE USER MAY USE ANY OF THESE STATE VARIABLES OR 018440
 C ANY OTHER SYSTEM PARAMETERS STORED IN COMMON TO CREATE THE 018441
 C EXTERNAL FORCE/TORQUE VECTORS REQUIRED BY EQUATION 11-74 OF 018442
 C VOL 1. 018443
 C 018444
 C THE USER MUST ADHERE TO THE FOLLOWING RULES IN ORDER TO CORRECTLY 018445
 C DEFINE THE ACTION OF EXTERNAL FORCES AND TORQUES: 018446
 C 1) EXTERNAL FORCES AND TORQUES MAY BE APPLIED ONLY AT 018447
 C 'SENSOR POINTS'. A MAXIMUM OF NSPMAX SUCH POINTS MAY BE 018448
 C DEFINED (VIA DATA INPUT IN DYNSTD) 018449
 C 2) DEFINE THE FOLLOWING ACRONYMS AS 018450
 C NTEX = TOTAL NUMBER OF 6-ELEMENT FORCE/TORQUE VECTORS 018451
 C TO BE DEFINED IN EXTOR AND RETURNED TO TORQUE. 018452
 C ISPN = FORCE/TORQUE APPLICATION POINT ARRAY. THE FORCE/ 018453
 C TORQUE VECTORS MUST BE NUMBERED AND CORRELATED 018454
 C WITH SENSOR POINTS. HENCE THE USER MUST DEFINE 018455
 C THE ELEMENTS OF ISPN ON THE FIRST PASS THROUGH 018456
 C EXTOR SUCH THAT 018457
 C ISPN(L) = SENSOR POINT NUMBER ASSOCIATED WITH 018458
 C THE POINT AT WHICH FORCE/TORQUE 018459
 C VECTOR NUMBER L, L=1,2,....,NTEX 018460
 C IS TO BE APPLIED 018461
 C TEX = FORCE/TORQUE VECTOR ARRAY. FOR COMPUTATIONAL 018462
 C EFFICIENCY THE USER SHOULD RETURN A MAXIMUM OF ONE 018463
 C FORCE/TORQUE VECTOR PER SENSOR POINT, HENCE IT IS 018464
 C DIMENSIONED (6,NSPMAX) IN TORQUE. THE ELEMENTS OF 018465
 C FORCE/TORQUE VECTOR L ARE TO BE STORED IN COLUMN L 018466
 C OF TEX ACCORDING TO THE FOLLOWING DEFINITIONS 018467
 C TEX(I,L), I=1,2,3 = X,Y,Z COMPONENTS OF THE 018468
 C RESULTANT EXTERNAL TORQUE L TO 018469
 C BE APPLIED AT SENSOR POINT 018470
 C ISPN(L), RELATIVE TO THE LOCAL 018471
 C ISPN(L) FIXED REFERENCE FRAME 018472
 C TEX(I,L), I=4,5,6 = X,Y,Z COMPONENTS OF THE 018473
 C RESULTANT EXTERNAL FORCE L TO 018474
 C BE APPLIED AT SENSOR POINT 018475
 C ISPN(L), RELATIVE TO THE LOCAL 018476
 C ISPN(L) FIXED REFERENCE FRAME 018477
 C BOTH DISTRIBUTED AND LOCALLY APPLIED EXTERNAL 018478
 C EFFECTS MAY BE CONSIDERED. THEY MAY BE ANY LINEAR 018479
 C OR NON-LINEAR FUNCTION OF PLANT AND CONTROL SYSTEM 018480
 C PARAMETERS; HOWEVER, ONLY RESULTANT FORCE/TORQUE 018481
 C VECTORS ACTING AT PARTICULAR SENSOR POINTS ARE TO 018482
 C BE RETURNED IN THE ARRAY TEX 018483
 C 3) UPON RETURNING TO TORQUE FROM EXTOR DISCUS MAKES USE OF 018484
 C THE DATA NOW CONTAINED IN THE ABOVE ARRAYS TO EVALUATE 018485

C	EQUATION 11-74 OF VOL I WITH NO ADDITIONAL USER INTERFACE	018486
C		018487
C		018488
C	SUBROUTINE KHINGE SPRINGS, DAMPERS AND MOTORS AT HINGES	018489
C	-----	018490
C		018491
C	RELATIVE MOTION OF CONTIGUOUS BODIES IS OFTEN RESTRAINED	018492
C	BY MECHANISMS PLACED AT THEIR COMMON HINGE POINT. THESE	018493
C	MECHANISMS, WHICH MAY INCLUDE LINEAR OR NON-LINEAR SPPINGS,	018494
C	DAMPERS AND MOTORS, PRODUCE INTERNAL FORCES AND TORQUES WHICH	018495
C	ACT IN EQUAL AND OPPOSITE PAIRS ON THE CONTIGUOUS BODIES. ALL	018496
C	SUCH EFFECTS MUST BE USER DEFINED IN SUBROUTINE KHINGE. (SEE	018497
C	PAGE III-6 OF VOL II).	018498
C		018499
C	KHINGE PROVIDES THE USER WITH A CLEAN INTERFACE BETWEEN PARAMETERS	018500
C	WHICH ARE PHYSICALLY REALIZABLE AND PARAMETERS REQUIRED FOR	018501
C	COMPUTATION.	018502
C		018503
C	RECALL THAT AT HINGE POINT L,L=1,2,....,NH	018504
C	EULER ANGLE ROTATION SEQUENCE TYPE IHDATA(1,L)	018505
C	IS USED TO ORIENT THE P-FRAME IN BODY ITOPOL(2,L)	018506
C	RELATIVE TO THE Q-FRAME IN BODY ITOPOL(1,L);	018507
C	FURTHERMORE DEGREE OF FREEDOM I,I=1,2,....6 IS	018508
C	FREE OF EITHER FIXED OR RHEONOMIC CONSTRAINTS IF	018509
C	IHDATA(I+1,L)=0.	018510
C		018511
C	TO PROVIDE THE USER WITH THE DESIREABLE CLEAN INTERFACE A	018512
C	A MODCLLING RESTRICTION MUST BE INTRODUCED; THAT IS,	018513
C		018514
C	INTERNAL TORQUE GENERATORS ARE ASSUMED TO	018515
C	ACT ABOUT EULER ANGLE ROTATION AXES	018516
C	WHICH ARE FREE OF FIXED OR	018517
C	RHEONOMIC CONSTRAINTS	018518
C		018519
C	THE USER MUST PROVIDE DISCOS WITH THE COMPONENTS OF THE	018520
C	RESULTANT INTERNALLY GENERATED TORQUE RELATIVE TO THE SKEWED	018521
C	AXIS SYSTEM WHICH WAS USER SPECIFIED IN ARRAY IHDATA.	018522
C	FURTHERMORE,	018523
C		018524
C	INTERNAL FORCE GENERATORS ARE ASSUMED TO	018525
C	ACT ALONG COORDINATE AXES OF THE	018526
C	Q-FRAME WHICH ARE FREE OF	018527
C	FIXED OR RHEONOMIC	018528
C	CONSTRAINTS	018529
C		018530
C	THE USER MUST PROVIDE DISCOS WITH THE COMPONENTS OF THE	018531
C	RESULTANT INTERNALLY GENERATED FORCE RELATIVE TO THE Q-FRAME	018532
C	COORDINATE AXES WHICH WAS USER SPECIFIED VIA INPUT DATA	018533
C		018534
C	THE USER MUST USE THE ACRONYM FNST TO DEFINE THE ARRAY IN WHICH	018535
C	THE TORQUE AND FORCE COMPONENTS ARE STORED AND ADHERE TO	018536
C	THE FOLLOWING DEFINITIONS:	018537
C	AT HINGE POINT L,L=1,2,....,NH	018538
C	FNST(1,L) = INTERNAL TORQUE TO BE APPLIED ABOUT FIRST EULER	018539
C	ROTATION AXIS AT HINGE POINT L	018540
C	FNST(2,L) = INTERNAL TORQUE TO BE APPLIED ABOUT SECOND EULER	018541
C	ROTATION AXIS AT HINGE POINT L	018542
C	FNST(3,L) = INTERNAL TORQUE TO BE APPLIED ABOUT THIRD EULER	018543
C	ROTATION AXIS AT HINGE POINT L	018544
C	FNST(4,L) = INTERNAL FORCE TO BE APPLIED ALONG THE X-AXIS OF	018545

```

C          THE Q-FRAME AT HINGE POINT L                                018546
C      FNOT(5,L) = INTERNAL FORCE TO BE APPLIED ALONG THE Y-AXIS OF    018547
C          THE Q-FRAME AT HINGE POINT L                                018548
C      FNOT(6,L) = INTERNAL FORCE TO BE APPLIED ALONG THE Z-AXIS OF    018549
C          THE Q-FRAME AT HINGE POINT L                                018550
C      FOR EXAMPLE ASSUME MOTION RELATIVE TO DEGREE OF FREEDOM        018551
C          I,I=1,2,....,6 AT HINGE POINT L,L=1,2,....,NM             018552
C          IS UNCONSTRAINED (IHDATA(I+1,L)=0) BUT THAT                018553
C          IT IS RESTRAINED BY A LINEAR SPRING OF                     018554
C          STIFFNESS SK(I,L) AND A LINEAR DAMPER HAVING              018555
C          A DAMPING CONSTANT DK(I,K). TO INCLUDE THIS               018556
C          IN DISCOS THE USER SIMPLY CODES                           018557
C      HNGT(I,L) = -(SK(I,L)*BETAH(I,L) + DK(I,L)*BETAHD(I,L))      018558
C          IN KHINGE. MOTOR GENERATED TORQUES (I=1,2,3)              018559
C          OR FORCES (I=4,5,6) ARE SIMILARLY HANDLED. IN            018560
C          MOST CASES THESE WOULD BE COMPUTED IN CONTRL              018561
C          AND PASSED TO KHINGE IN A SPECIAL COMMON                  018562
C          BLOCK. IF TQFR(I,L) IS THE MOTOR TORQUE OR                018563
C          FORCE RESTRAINING DEGREE OF FREEDOM I MOTION              018564
C          AT HINGE POINT L ITS EFFECT WOULD BE INCLUDED             018565
C          BY THE USER WRITTEN CODE                                  018566
C      HNGT(I,L) = HNGT(I,L) + TQFR(I,L)                             018567
C                                                                    018568
C      ALL DATA REQUIRED BY THE USER IN KHINGE; SUCH AS, SPRING AND  018569
C      DAMPING CONSTANTS, ARE MOST CONVENIENTLY INPUTTED IN ARRAY   018570
C      CNTDTA. TIME DEPENDENT DATA REQUIRED BY THE USER; SUCH AS,   018571
C      MOTOR TORQUES OR FORCES, ARE MOST CONVENIENTLY COMPUTED IN   018572
C      KHINGE OR COMPUTED IN CONTRL AND PASSED TO KHINGE BY A SPECIAL 018573
C      PURPOSE COMMON BLOCK                                          018574
C                                                                    018575
C      IF MECHANISMS EXIST IN THE SYSTEM WHICH CAN STORE POTENTIAL  018576
C      THEIR CONTRIBUTION TO TOTAL SYSTEM POTENTIAL ENERGY (TOTPE)  018577
C      MUST BE ACCOUNTED FOR. FOR THE LINEAR SPRING ASSUMED ABOVE    018578
C      ITS CONTRIBUTION IS INCLUDED BY THE USER WRITTEN CODE        018579
C          TOTPE = TOTPE + .500*SK(I,L)*BETAH(I,L)**2              018580
C      CONTRIBUTIONS DUE TO NON-LINEAR SPRINGS MUST BE USER         018581
C      DETERMINED AND ALSO CODED. TOTAL SYSTEM POTENTIAL ENERGY IS 018582
C      NOT USED FOR ANY REQUIRED COMPUTATION IN DISCOS               018583
C                                                                    018584
C      UPON ENTRY INTO KHINGE CODE IS PROVIDED TO ZERO TOTPE AND ALL 018585
C      ELEMENTS OF THE ARRAY HNGT. THE USER THEN PROVIDES THE CODE   018586
C      TO DEFINE THE NON-ZERO ELEMENTS OF HNGT AND THE ELEMENT TOTPE. 018587
C      DISCOS PROCEEDS THEN VIA THE PROVIDED CODED ALGORITHM TO     018588
C      COMPUTE PARAMETERS REQUIRED FOR CONTINUING COMPUTATION.        018589
C                                                                    018590
C                                                                    018591
C      SUBROUTINE SHAFT   WHEEL TORQUES                               018592
C      -----                                                        018593
C                                                                    018594
C      VARIABLE AND CONSTANT SPEED WHEELS MAY BOTH BE INCLUDED IN THE 018595
C      SIMULATION MODEL. WHEEL TORQUES MAY BE APPLIED ONLY TO THE   018596
C      VARIABLE SPEED WHEELS OF THE SYSTEM. THIS IS DONE IN A MANNER 018597
C      ALMOST IDENTICAL TO THAT USED IN SUBROUTINE KHINGE.          018598
C                                                                    018599
C      ONLY INTERNALLY GENERATED TORQUES MAY BE APPLIED TO WHEELS, AND 018600
C      THESE ACT ONLY ABOUT THE RESPECTIVE WHEEL AXES.              018601
C                                                                    018602
C          WHEEL I,I=1,2,....,NCFMU IS A VARIABLE SPEED WHEEL      018603
C          IF(IMD(3,I).EQ.1)                                          018604
C          WHERE THE IMD DATA ARRAY IS USER SPECIFIED VIA          018605

```

C	INPUT DATA IN DYN510	018606
C		018607
C	IN SUBROUTINE SHAFT THE USER MUST USE THE ACRONYM TSHT TO DEFINE	018608
C	THE ARRAY IN WHICH ALL WHEEL TORQUES ARE STORED AND ADHERE TO	018609
C	THE FOLLOWING DEFINITION FOR ELEMENT I OF THE ARRAY TSHT	018610
C	WHICH IS DIMENSIONED (NMWMAX) IN SUBROUTINE TORQUE	018611
C		018612
C	TSHT(I) = INTERNALLY GENERATED TORQUE TO BE APPLIED ABOUT	018613
C	THE SPIN AXIS OF WHEEL I. AS IN KHINGE THIS	018614
C	TORQUE IS USUALLY COMPUTED IN CONTRL AND PASSED	018615
C	TO SHAFT VIA A SPECIAL PURPOSE COMMON BLOCK	018616
C		018617
C	THE COMPUTATION OF WHEEL TORQUE OFTEN REQUIRES KNOWLEDGE OF THE	018618
C	INSTANTANEOUS SPIN RATE AND AT TIMES, FOR SPECIAL SIMULATION	018619
C	PROBLEMS, WHEEL ANGLE. ALL SPIN RATES OF VARIABLE SPEED	018620
C	(ACTIVE) WHEELS ARE CONTAINED IN THE STATE VECTOR ARRAY Y, SEE	018621
C	DEFINITION ABOVE FOR EXACT LOCATION. SINCE WHEEL RATES ARE	018622
C	OFTEN VERY HIGH AND WHEEL ANGLE IS USUALLY NOT REQUIRED FOR	018623
C	SIMULATION PURPOSES WHEEL ANGLE IS NOT COMPUTED. IF REQUIRED,	018624
C	HOWEVER THE USER SIMPLY HAS TO ADD THE DIFFERENTIAL EQUATION	018625
C	IN CONTRL WHICH WILL COMPUTE IT.	018626
C		018627
C	FOR EXAMPLE: IF THE SPIN OF WHEEL I IS DAMPED BY A LINEAR DAMPER	018628
C	WITH DAMPING CONSTANT WDK(I) AND CONTROLLED BY A	018629
C	MOTOR WHICH PRODUCES WHEEL TORQUE CLM(I) THE	018630
C	USER MUST CODE	018631
C	IW = LOCATION IN ARRAY Y OF WHEEL I SPIN RATE. (A SIMPLE	018632
C	INTEGER EQUATION BASED UPON ABOVE DEFINITION)	018633
C	TSHT(I) = -WDK(I)*Y(IW) + CLM(I)	018634
C		018635
C		018636
C	SUBROUTINE GMISC MISCELLANEOUS AND THERMAL EFFECTS	018637
C	-----	018638
C		018639
C	THERMALLY INDUCED MOTION OF A FLEXIBLE BODY CAN OCCUR IF THERMAL	018640
C	TIME CONSTANTS AND FUNDAMENTAL NATURAL FREQUENCIES ARE OF	018641
C	NEARLY THE SAME ORDER OF MAGNITUDE. THE MOTION CAN AT TIMES	018642
C	BE SELF-SUSTAINING AND EVEN UNSTABLE IF THE DIRECTION AND	018643
C	MAGNITUDE OF THE INCIDENT THERMAL FIELD IS TIME VARYING	018644
C	RELATIVE TO A BODY FIXED REFERENCE. IN GENERAL THE FLEXIBLE	018645
C	BODY WILL MOVE TOWARD A TIME VARYING POSITION OF THERMAL	018646
C	EQUILIBRIUM WHICH WILL BE DEPENDENT UPON THE ORIENTATION OF	018647
C	THE BODY RELATIVE TO THE THERMAL FIELD	018648
C		018649
C	THIS EFFECT CAN BE INCORPORATED IN SUBROUTINE GMISC. IN GMISC	018650
C	THE USER MUST HAVE AT HAND THE TIME VARYING MODAL AMPLITUDES	018651
C	OF THE THERMAL EQUILIBRIUM POSITION, USUALLY OBTAINED VIA	018652
C	SOLUTION OF HEAT CONDUCTION EQUATIONS IN CONTRL. THE POSITION	018653
C	OF THERMAL EQUILIBRIUM IS THE POSITION THAT ELASTIC RESTORING	018654
C	FORCES ARE MEASURED FROM. THIS COMPUTATION IS CARRIED OUT	018655
C	IN GMISC. THIS IS AN ADVANCED TOPIC; IT SUFFICES TO SAY THAT	018656
C	DISCOS CAN BE SETUP TO INCLUDE THIS EFFECT IF REQUIRED.	018657
C		018658
C		018659
C	SUBROUTINE EQADD SENSOR AND TORQUE SIGNAL EQUATIONS	018660
C	-----	018661
C		018662
C	THE CONTROL SYSTEM DESIGNER NEEDS THE ABILITY TO OBTAIN THE	018663
C	TRANSFER FUNCTIONS WHICH RELATE PHYSICALLY REALIZABLE SENSOR	018664
C	SIGNALS, WHICH ARE FUNCTIONS OF PLANT STATE VARIABLES, TO	018665

C	PHYSICALLY REALIZABLE TORQUE SIGNALS, WHICH ARE FUNCTIONS	018666
C	OF CONTROL SYSTEM STATE VARIABLES.	018667
C		018668
C	TO ACCOMPLISH THIS THE USER MUST DEFINE A SET OF NAUX AUXILIARY	018669
C	EQUATIONS. IN SUBROUTINE CNTRL THE USER DEFINED THAT IN	018670
C	SUBROUTINE EQADD THERE WOULD BE CODED EXACTLY	018671
C	NXSS SENSOR SIGNAL EQUATIONS, AND EXACTLY	018672
C	NBTO TORQUE SIGNAL EQUATIONS	018673
C	SUCH THAT	018674
C	NAUX = NXSS+NBTO	018675
C		018676
C	AS IN ALL OTHER USER DEFINED ROUTINES CERTAIN RULES MUST BE	018677
C	ADHERED TO BY THE USER, THESE ARE AS FOLLOWS:	018678
C	1) SENSOR SIGNALS - THE USER MUST DEFINE NXSS SENSOR SIGNAL	018679
C	EQUATIONS, THESE EQUATIONS ARE ASSUMED	018680
C	TO BE FUNCTIONS OF PLANT STATE VARIABLES.	018681
C	IN THE PROCESS OF NUMERICALLY LINEARIZING	018682
C	EQUATIONS ALL SENSOR SIGNAL EQUATIONS	018683
C	WILL BE EXPRESSED AS LINEAR FUNCTIONS OF	018684
C	PLANT STATE VARIABLES	018685
C	THE USER CODES IN EQADD	018686
C	YOT(NEQ+I) = I-TH SENSOR SIGNAL EQUATION	018687
C	I=1,2,...,NXSS	018688
C	2) TORQUE SIGNALS - THE USER MUST DEFINE NBTO TORQUE SIGNAL	018689
C	EQUATIONS, THESE EQUATIONS ARE ASSUMED	018690
C	TO BE FUNCTIONS OF CONTROL SYSTEM STATE	018691
C	VARIABLES. THE USER MUST BE CAREFUL TO	018692
C	INCLUDE CONTROL EQUATIONS IN CNTRL WHICH	018693
C	CAN BE USED TO DEFINE ALL TORQUE SIGNALS.	018694
C	THIS IS NECESSARY BECAUSE IN THE	018695
C	NUMERICAL LINEARIZATION PROCESS ALL	018696
C	TORQUE SIGNALS WILL BE EXPRESSED AS	018697
C	LINEAR FUNCTIONS OF THE CONTROL SYSTEM	018698
C	STATE VARIABLES	018699
C	THE USER CODES IN EQADD	018700
C	YOT(NEQ+NXSS+J) = J-TH TORQUE SIGNAL EQUATION	018701
C	J=1,2,...,NBTO	018702
C		018703
C		018704
C	SYSTEM AND BODY ENERGY AND MOMENTUM	018705
C	-----	018706
C		018707
C	THE FOLLOWING PARAMETERS ARE CONTINUALLY MONITORED FOR THE USER	018708
C	TO CHECK TO DETERMINE IF ALL ENERGY AND MOMENTUM CONSERVATION	018709
C	LAWS ARE HOLDING TRUE. THESE PARAMETERS ARE WITH THE EXCEPTION	018710
C	OF ORDINARY MOMENTA COMPUTED IN SUBROUTINE ENGMOM, THEY ARE	018711
C	ALL STORED IN /MOMENG/ AND PRINTED AS A FUNCTION OF TIME	018712
C		018713
C	P = ORDINARY MOMENTA ARRAY, SEE EQUATION 11-49 VOL 1	018714
C	DIMENSIONED (NBMAX*(6+NMDOD)+NM*MAX)	018715
C	(P(I),I=1,NU) = ORDINARY MOMENTA ASSOCIATED WITH VELOCITY	018716
C	COORDINATE I, I=1,...,NU	018717
C	FMDM = BODY ANGULAR AND LINEAR MOMENTUM ARRAY	018718
C	DIMENSIONED (6*NBMAX)	018719
C	(FMDM(I),I=6*(N-1)+1,6*(N-1)+3) = 3 COMPONENTS OF THE INERTIAL	018720
C	ANGULAR MOMENTUM VECTOR ASSOCIATED WITH THE	018721
C	MOTION OF BODY N WRT INERTIAL COORDINATES	018722
C	(FMDM(I),I=6*(N-1)+4,6*(N-1)+6) = 3 COMPONENTS OF THE INERTIAL	018723
C	LINEAR MOMENTUM VECTOR ASSOCIATED WITH THE	018724
C	MOTION OF BODY N WRT INERTIAL COORDINATES	018725

```

C      HTOT   = TOTAL SYSTEM ANGULAR MOMENTUM ARRAY DIMENSIONED (3) THE 3 018726
C              COMPONENTS ARE WRT INERTIAL COORDINATES 018727
C      TOTL   = TOTAL SYSTEM LINEAR MOMENTUM ARRAY DIMENSIONED (3) THE 3 018728
C              COMPONENTS ARE WRT INERTIAL COORDINATES 018729
C      ENGKE  = KINETIC ENERGY ARRAY, DIMENSIONED (NUMAX) 018730
C              ENGKE(N) = KINETIC ENERGY OF BODY N 018731
C      ENGPE  = POTENTIAL ENERGY ARRAY, DIMENSIONED (NUMAX) 018732
C              ENGPE(N) = POTENTIAL ENERGY OF BODY N 018733
C      TOTKE  = TOTAL SYSTEM KINETIC ENERGY 018734
C      TOTPE  = TOTAL SYSTEM POTENTIAL ENERGY 018735
C      ATOTL  = TOTAL SYSTEM LINEAR MOMENTUM (VECTOR MAGNITUDE) 018736
C      AHTOT  = TOTAL SYSTEM ANGULAR MOMENTUM (VECTOR MAGNITUDE) 018737
C 018738
C 018739
C      LINEAR = LINEARIZED SYSTEM EQUATIONS OF MOTION 018740
C----- 018741
C 018742
C      THE LINEARIZATION PROCESS PROVIDES A SET OF LINEAR COUPLED FIRST 018743
C      ORDER DIFFERENTIAL EQUATIONS WHICH DEFINE THE TOTAL SYSTEM 018744
C      RESPONSE RELATIVE TO AN EQUILIBRIUM STATE. 018745
C 018746
C      IF THE COUPLED NON-LINEAR EQUATIONS OF MOTION ARE TO BE LINEARIZED 018747
C      THE INITIAL STATE DEFINED BY THE INPUTTED SYSTEM INITIAL 018748
C      CONDITIONS DEFINES THE EQUILIBRIUM STATE RELATIVE TO WHICH 018749
C      PERTURBATION EQUATIONS ARE TO BE DERIVED 018750
C 018751
C      LET: 018752
C      (Y(I,0),I=1,NEQ) = EQUILIBRIUM STATE OF THE STATE VECTOR 018753
C                      DEFINED BY USER SPECIFIED INITIAL 018754
C                      CONDITIONS 018755
C      (Y(I,T),I=1,NEQ) = STATE VECTOR AT TIME T 018756
C      THEN: 018757
C 018758
C      (DLY(I,T) = Y(I,T)-Y(I,0),I=1,NEQ) 018759
C 018760
C      IS THE PERTURBATION VECTOR 018761
C 018762
C      SIMILARLY LET: 018763
C      (YDT(I,0),I=1,NEQ) = EQUILIBRIUM STATE OF THE FIRST TIME 018764
C                      DERIVATIVE OF THE STATE VECTOR 018765
C      (YDT(I,T),I=1,NEQ) = STATE RATE VECTOR AT TIME T 018766
C      (YDT(I,0),I=NEQ+1,NAUX) = EQUILIBRIUM STATE FOR ALL AUXILARY 018767
C                      EQUATIONS DEFINED BY THE USER IN 018768
C                      ECADD, THESE ARE SENSOR AND TORQUE 018769
C                      SIGNAL EQUATIONS FOR TRANSFER 018770
C                      FUNCTION STUDIES 018771
C      (YDT(I,T),I=NEQ+1,NAUX) = SENSOR AND TORQUE SIGNALS AT TIME T 018772
C      THEN: 018773
C 018774
C      (DLYDT(I,T) = YDT(I,T)-YDT(I,0),I=1,NEQ+NAUX) 018775
C 018776
C      IS THE PERTURBATION RATE VECTOR 018777
C 018778
C      THE NUMERICALLY OBTAINED LINEARIZED PERTURBATION EQUATIONS 018779
C      TO BE OBTAINED VIA SUBROUTINE LINEAR ARE: 018780
C 018781
C      DLYDT(I,T) = H(I,J)*DLY(J,T) ; I=1,2,...,NEQ+NAUX 018782
C 018783
C      WHERE SUMMATION IS IMPLIED BY THE REPEATED J INDEX 018784
C      J=1,2,...,NEQ 018785

```

```

C      THE ELEMENTS OF THE NEG+NAUX BY NEG PARTIAL DERIVATIVE      018786
C      MATRIX 'H' ARE OBTAINED A COLUMN AT A TIME                  018787
C                                                                    018788
C      ONLY THOSE COLUMNS J OF THE PARTIAL DERIVATIVE MATRIX 'H' 018789
C      CORRESPONDING TO INDEPENDENT COORDINATES ARE REQUIRED; THAT IS 018790
C      ONLY THOSE COLUMNS J FOR WHICH INDEP(J)=1, J=1,2,...,NEG 018791
C      ARE EVALUATED. A MAXIMUM OF 30 ITERATION STEPS ARE ALLOWED 018792
C      FOR THE CONVERGENCE OF EACH COLUMN. IF DEBUG(40)=.TRUE. THE 018793
C      FOLLOWING PRINTOUT IS OBTAINED                               018794
C      * Z      ARRAY ITERATION N FOR COLUMN J * = THE NEG+NAUX ELEMENTS IN 018795
C      COLUMN J OF THE PARTIAL DERIVATIVE MATRIX AT                018796
C      THE START OF ITERATION N                                     018797
C      * ZNEW ARRAY ITERATION N FOR COLUMN J * = THE NEG+NAUX ELEMENTS IN 018798
C      COLUMN J OF THE PARTIAL DERIVATIVE MATRIX AT                018799
C      THE END OF ITERATION N                                       018800
C      FOR CONVERGENCE THE FOLLOWING CODED TEST IS MADE            018801
C      DO 100 I=1,NEG+NAUX                                          018802
C      IF(INDEP(I).EQ.0) GO TO 100                                    018803
C      DN = DABS(Z(I))                                              018804
C      DNI = DABS(ZNEW(I))                                          018805
C      IF(DNI.GT.DN) DN=DNI                                         018806
C      IF(DN.LT.EPS1) GO TO 100                                     018807
C      G1 = DABS(ZNEW(I)-Z(I))/DN                                   018808
C      IF(G1.LE.EPS2) GO TO 100                                    018809
C      C      SORRY TRY AGAIN; ONLY 30 CHANCES ALLOWED              018810
C      C      TO ACHIEVE CONVERGENCE FOR ALL COMPONENTS            018811
C      C      100 CONTINUE                                          018812
C      C      CONVERGENCE TEST SUCCESSFUL                            018813
C      C      EPS1 = 1.00-08      THESE NUMBERS MAY BE PROBLEM    018814
C      C      EPS2 = 1.00-04      SENSITIVE                         018815
C      ** THIS CRITERIA HAS BEEN SLIGHTLY MODIFIED TO PROVIDE      018816
C      FOR TWO AUTOMATIC CHANGES OF EPS1. FOR COLUMNS WITH VERY 018817
C      LARGE MAGNITUDE NUMBERS EPS1 = 1.00-03 IS BEYOND THE        018818
C      ACCURACY CAPABILITIES OF THE COMPUTER. THE CRITERIA FOR     018819
C      RESETTING IS BASED UPON THE LARGEST MAGNITUDE NUMBER        018820
C      IN THE Z ARRAY FOUND ON THE FIRST ITERATION THROUGH.        018821
C      IF EPS1 = 1.00-08 IS UNSUCCESSFUL                           018822
C      RESET EPS1 = 1.00-06*MAX(Z(I) OF ITERATION 1) AND           018823
C      THEN TO 1.00-04*MAX(Z(I) OF ITERATION 1) BEFORE GIVING UP 018824
C                                                                    018825
C      WHEN SUCCESSFUL CONVERGENCE OBTAINED THE FOLLOWING PRINTOUT 018826
C      IS PROVIDED                                                  018827
C      * COLUMN J OF THE PARTIAL DERIVATIVE MATRIX IS * = THE NJQ ELEMENTS 018828
C      IN COLUMN J OF THE PARTIAL DERIVATIVE MATRIX                018829
C      WHICH MUST BE SAVED FOR LINEAR ANALYSIS. THESE              018830
C      ARE THE ELEMENTS OF ARRAY ZNEW CORRESPONDING TO              018831
C      EITHER INDEPENDENT COORDINATES OR SENSOR AND                018832
C      TORQUE SIGNALS. (THIS COLUMN MATRIX IS STORED               018833
C      ON TAPE2 FOR RETRIEVAL IN DYN540                             018834
C      NOTE: NJQ = NX + NAUX                                         018835
C                                                                    018836
C                                                                    018837
C                                                                    018838
C                                                                    018839
C                                                                    018840
C                                                                    018841
C      STABILITY ANALYSIS OF LINEARIZED EQUATIONS, DYN540          018842
C      *****                                                      018843
C                                                                    018844
C      ** NOTE **      STABILITY ANALYSIS OF LARGE ORDER SYSTEMS, WHICH 018845

```

```

C          IS WHAT DISCUS IS GEARED FOR, REQUIRES OPERATING 018846
C          WITH LARGE ORDER MATRICES. TO CONSERVE MEMORY 018847
C          REQUIREMENTS, WORK AREAS AND SCRATCH FILES ARE 018848
C          EXTENSIVELY USED. BY SETTING THE APPROPRIATE 018849
C          DEBUG FLAG EQUAL .TRUE., PARTICULAR MATRICES CAN 018850
C          BE PRINTED. IN MOST CASES THEY ARE DESTROYED AS 018851
C          SOON AS THEY ARE NO LONGER NEEDED. 018852
C 018853
C 018854
C          LINEARIZED EQUATIONS OF MOTION 018855
C          ----- 018856
C 018857
C          THE STABILITY ANALYSIS IS DIRECTED BY SUBROUTINE DYN540. IN 018858
C          SUBROUTINE LINEAR THE COUPLED SYSTEM EQUATIONS ALONG WITH THE 018859
C          SENSOR AND TORQUE SIGNAL EQUATIONS ARE LINEARIZED AND THE 018860
C          PARTIAL DERIVATIVE MATRIX IS STORED FOR RECOVERY IN DYN540 ON 018861
C          TAPE2. UPON ENTRY INTO DYN540 THIS MATRIX IS READ, STORED 018862
C          AND PRINTED. THE EQUATIONS ARE OF THE FORM 018863
C 018864
C          DLYDT(I,T) = H(I,J)*DLY(J,T) 018865
C 018866
C          WHERE: 018867
C          I=1,....,NX-NDELTA INDICES ASSOCIATED WITH INDEPENDENT PLANT 018868
C          COORDINATES 018869
C          I=NX-NDELTA+1,....,NX INDICES ASSOCIATED WITH INDEPENDENT 018870
C          CONTROL SYSTEM COORDINATES 018871
C          I=NX+1,....,NX+NXSS INDICES ASSOCIATED WITH SENSOR SIGNALS 018872
C          AS DEFINED IN EQA00 018873
C          I=NX+NXSS+1,....,NJQ INDICES ASSOCIATED WITH TORQUE SIGNALS 018874
C          AS DEFINED IN EQA00 018875
C          AND THE REPEATED INDEX J IMPLIES SUMMATION OVER J; J=1,....,NX 018876
C          WORK ARRAYS W1 AND W2 IN /LWORK1/ ARE USE AS TEMPORARY STORAGE 018877
C          LOCATIONS FOR PARTITIONS OF THE H MATRIX. THE SYMBOLS DLY AND 018878
C          DLYDT ARE DEFINED ABOVE UNDER THE TITLE 'LINEAR - ...' 018879
C 018880
C          THE OUTPUT MATRICES PRINTED AFTER THE FOLLOWING CAPTIONS ARE: 018881
C          * OUTPUT MATRIX -A- (NX X NX)' = ((H(I,J),J=1,NX),I=1,NX); 018882
C          THAT IS, THE COEFFICIENTS OF THE PARTIAL 018883
C          DERIVATIVE MATRIX ASSOCIATED WITH THE 018884
C          INDEPENDENT PLANT AND CONTROL SYSTEM STATE 018885
C          VARIABLES (THESE ARE DEFINED BY ARRAY INDEF) 018886
C          * OUTPUT MATRIX -A-AUX (NAUX X NX)' = ((H(I,J),J=1,NX),I=1,NAUX); 018887
C          THAT IS, THE COEFFICIENTS OF THE PARTIAL 018888
C          DERIVATIVE MATRIX ASSOCIATED WITH THE EQUATIONS 018889
C          WHICH DEFINE SENSOR AND TORQUE SIGNALS AS 018890
C          LINEAR FUNCTIONS OF PLANT AND CONTROL SYSTEM 018891
C          STATE VARIABLES RESP. 018892
C 018893
C          ** NOTE ** THE CONTENTS OF THIS MATRIX ARE PARTICULARLY 018894
C          USEFUL WHEN THE HIGHLY COUPLED SYSTEMS ARE 018895
C          MODELLED. THE RELATIVE MAGNITUDES OF THE ELEMENTS 018896
C          IN ANY ROW PROVIDE A MEASURE OF THE PARTICULAR 018897
C          SENSOR OR TORQUE SIGNAL'S DEPENDENCE ON EACH OF 018898
C          THE SYSTEM'S INDEPENDENT DEGREES OF FREEDOM IN 018899
C          THE TIME DOMAIN 01900
C 01901
C 01902
C          SIMILARITY TRANSFORMATION 01903
C          ----- 01904
C 01905

```

```

C      TO OBTAIN TRANSFER FUNCTIONS BETWEEN SELECTED SENSOR SIGNALS AND      018906
C      SELECTED TORQUE SIGNALS A SIMILARITY TRANSFORMATION IS MADE          018907
C      WHICH NUMERICALLY SELCTS IN AN OPTIMUM MANNER EXACTLY NAUX          018908
C      INDEPENDENT SYSTEM STATE VARIABLES FROM THE STATE VECTOR           018909
C      AND REPLACES THEM WITH THE NAUX SENSOR AND TORQUE SIGNAL             018910
C      VARIABLES. THE RESULTING EQUATION III-24 OF VOL 1 IS OF THE           018911
C      FORM: (THIS OPERATION IS DONE IN ASIMLR AND FINDT)                   018912
C                                                                            018913
C       $ZDT(I,T) = R^{**}(-1)*H*R(I,J) * Z(J,T)$                                018914
C                                                                            018915
C      WHERE:                                                                018916
C      (I,I=1,NY2) = INDICES OF THE PERTURBATION VARIABLES                018917
C                   ASSOCIATED WITH THE PLANT WHICH ARE NOT               018918
C                   ELIMINATED BY THE SIMILARITY TRANSFORMATION            018919
C      (I,I=NY2+1, = INDICES OF THE PERTURBATION VARIABLES                018920
C                   NY2+NXSS) ASSOCIATED WITH THE NXSS SENSOR SIGNAL        018921
C                   VARIABLES                                               018922
C      (I,I=NX-NDELTA, = INDICES OF THE PERTURBATION VARIABLES            018923
C                   NX-NSTQ) ASSOCIATED WITH THE CONTROL SYSTEM WHICH      018924
C                   ARE NOT ELIMINATED BY THE SIMILARITY TRANSFORMATION    018925
C      (I,I=NX-NBTQ+1, = INDICES OF THE PERTURBATION VARIABLES            018926
C                   NX) ASSOCIATED WITH THE NBTQ TORQUE SIGNAL             018927
C                   VARIABLES                                               018928
C      Z(I,T) = PERTURBATION VARIABLE I AT TIME T REFER TO 'THE TRANSFORMED 018929
C                   STATE VECTOR CORRELATION ARRAY' PRINTED TO CORRELATE 018930
C                   INDEX I WITH ORIGINAL STATE VARIABLE INDEXING STORED IN 018931
C                   LEND AND LOCU                                           018932
C      ZDT(I,T) = FIRST TIME DERIVATIVE OF Z(I,T)                         018933
C                   (INDEXING CONSISTANT WITH Z(I,T))                     018934
C      R**(-1)*H*R(I,J) = THE (I,J) ELEMENT OF THE SIMILARITY            018935
C                   TRANSFORMATION MATRIX (OUTPUT OF ASIMLR)              018936
C                   ORDERED AS JUST DEFINED ABOVE                        018937
C                                                                           018938
C      THE OUTPUT MATRICES PRINTED AFTER THE FOLLOWING CAPTIONS ARE        018939
C      ' OUTPUT MATRIX -A*- (NX X NX) = (R**(-1)*H*R(I,J),J=1,NX),        018940
C      I=1,NX); THAT IS THE ELEMENTS OF THE SIMILARITY TRANSFORMATION      018941
C      MATRIX COMPUTED IN ASIMLR AND PASSED BACK TO DYN540 IN THE ARRAY W2 018942
C      'RT A' = A LISTING IN DECREASING ORDER OF REAL PART OF ALL        018943
C      'NO REAL PART IMAGINARY PART' NX COMPLEX ROOTS OF THE              018944
C      PARTIAL DERIVATIVE MATRIX 'H' DEFINED ABOVE                       018945
C      'RTA*' = A LISTING IN DECREASING ORDER OF REAL PART OF ALL        018946
C      'NO REAL PART IMAGINARY PART' NX COMPLEX ROOTS OF THE              018947
C      SIMILARITY TRANSFORMATION MATRIX R**(-1)*H*R DEFINED ABOVE        018948
C                                                                           018949
C      ** NOTE **                                                           018950
C      BOTH SETS OF COMPLEX ROOTS SHOULD BE EQUAL. THIS IS A              018951
C      CHECK ON THE QUALITY OF THE SIMILARITY TRANSFORMATION              018952
C                                                                           018953
C      COMPLEX ROOTS VIA THE QR-ALGORITHM                                  018954
C                                                                           018955
C                                                                           018956
C                                                                           018957
C                                                                           018958
C                                                                           018959
C                                                                           018960
C                                                                           018961
C                                                                           018962
C                                                                           018963
C                                                                           018964
C                                                                           018965

```



```

C ----- 018966
C 018967
C COMPLEX ROOTS OF GENERAL NON-SYMMETRIC MATRICES OF REAL NUMBERS 018968
C ARE OBTAINED VIA THE QR-ALGORITHM. QRRVR ACCEPTS THE N BY N 018969
C MATRIX Q AND RETURNS TO THE CALLING ROUTINE THE COMPLEX ROOTS 018970
C IN THE ARRAYS NR (REAL ROOTS) AND RI (IMAGINARY ROOTS). THE 018971
C ROOTS ARE ARRANGE IN DECREASING ORDER OF REAL PART. 018972
C 018973
C IF DEBUG(64) = .TRUE. THE FOLLOWING CAPTIONS AND ASSOCIATED 018974
C MATRICES WILL BE PRINTED: 018975
C * ELEMENTS OF ARRAY Q' = THE N BY N INPUT MATRIX FOR WHICH THE 018976
C COMPLEX ROOTS ARE TO BE OBTAINED 018977
C * UPPER HESSENBERG FORM = THE Q-MATRIX IS PASSED TO SUBROUTINE 018978
C OF Q OUT OF SUBDIA IS' SUBDIA IN WHICH IT IS TRANSFORMED INTO 018979
C UPPER HESSENBERG FORM (ALMOST 018980
C TRIANGULAR) IN SO DOING THE ORIGINAL 018981
C MATRIX IS DESTROYED. DONE TO INCREASE 018982
C SPEED AND EFFICIENCY OF QR-ALGORITHM 018983
C * REAL ROOTS OUT OF = THE REAL PARTS OF THE N COMPLEX ROOTS 018984
C QRCN ARE' OF THE INPUT MATRIX Q IN DECREASING 018985
C ORDER 018986
C * IMAGINARY ROOTS OUT OF = THE ASSOCIATED IMAGINARY PARTS OF THE 018987
C QRCN ARE' N COMPLEX ROOTS, RESPECTIVELY 018988
C 018989
C 018990
C ASIMLR & FINDI - PERFORM THE SIMILARITY TRANSFORMATION 018991
C ----- 018992
C 018993
C DISCOS HAS THE UNIQUE FEATURE THAT TRANSFER FUNCTIONS MAY BE 018994
C OBTAINED BETWEEN PHYSICALLY REALIZABLE INPUT SENSOR SIGNALS 018995
C AND PHYSICALLY REALIZABLE TORQUE SIGNALS. THE SENSOR AND 018996
C TORQUE SIGNALS ARE USER DEFINED FUNCTIONS SPECIFIED IN EQADD. 018997
C TO ACHIEVE THIS CAPABILITY A SIMILARITY TRANSFORMATION MUST 018998
C BE DEVELOPED WHICH WILL ALLOW FOR AN EXCHANGE OF VARIABLES; 018999
C THAT IS, NXSS PLANT STATE VARIABLES ARE ELIMINATED FROM THE 019000
C STATE VECTOR AND REPLACED BY NXSS SENSOR SIGNAL VARIABLES. 019001
C SIMILARLY NBTQ CONTROL STATE VARIABLES ARE ELIMINATED AND 019002
C REPLACED WITH NBTQ TORQUE SIGNAL VARIABLES. (SEE PAGE III-7 019003
C OF VOL 1). THE MATRICES COMPUTED IN THESE ROUTINES ARE ONLY 019004
C STORED TEMPORARLY IN THE WORK ARRAYS W1 AND W2 IN /LWORK1/ 019005
C 019006
C BY SETTING DEBUG(65) AND DEBUG(66)=.TRUE. THE STEPS TAKEN THROUGH 019007
C ASIMLR AND FINDI TO ARRIVE AT THE SIMILARITY TRANSFORMATION 019008
C CAN BE FOLLOWED FOR ANY PARTICULAR PROBLEM. THE FOLLOWING 019009
C CAPTIONS AND ASSOCIATED MATRICES ARE PRINTED: 019010
C * INPUT MATRIX COEFFICIENTS = THE NAUX BY NX MATRIX DERIVED IN 019011
C FOR LINEARIZED SIGNALS' LINEAR WHICH CAN BE USED TO DEFINE 019012
C ALL SENSOR AND TORQUE SIGNALS AS 019013
C LINEAR FUNCTIONS OF INDEPENDENT 019014
C PLANT AND CONTROL SYSTEM STATE 019015
C VARIABLES. THIS IS THE INPUT MATRIX 019016
C TO ASIMLR. IT IS THE PARTITION 'C' 019017
C IN EQUATION III-26 019018
C * SUBROUTINE FINDI INPUT = THE NAUX BY NJQ COEFFICIENT MATRIX 019019
C MATRIX' IN EQUATION III-26. FINDI USES THIS 019020
C MATRIX TO FIND THE BEST SET OF 019021
C STATE VARIABLES TO BE REPLACED BY 019022
C SENSOR AND TORQUE SIGNAL VARIABLES 019023
C * VARIABLE EXCHANGE MATRIX' = THE NAUX BY NJQ VARIABLE EXCHANGE 019024
C MATRIX ILLUSTRATED PAGE 63. 019025

```

C		VOL 1. THE EXISTANCE OF 1.000D+00	019026
C		IN COLUMN J OF ROW 1 IMPLIES THAT	019027
C		THE J-TH INDEPENDENT STATE VARIABLE	019028
C		WILL BE NOT SURVIVE THE EXCHANGE	019029
C		PROCESS. THE STATE VARIABLES TO BE	019030
C		DELETED ARE THOSE WHICH THE SENSOR	019031
C		AND TORQUE SIGNALS ARE MOST	019032
C		STRONGLY DEPENDENT UPON. THESE ARE	019033
C		DETERMINED NUMERICALLY BY AN	019034
C		ALGORITHM WHICH ASSUMES THAT THE	019035
C		SIGNALS ARE MUTUALLY INDEPENDENT.	019036
C	' SIMILARITY TRANSFORMATION =	THE NX BY NX SIMILARITY	019037
C	MATRIX IS'	TRANSFORMATION MATRIX. IT IS	019038
C		CONSTRUCTED DIRECTLY FROM THE	019039
C		VARIABLE EXCHANGE MATRIX AND	019040
C		RETURNED TO ASIMLR IN THE DUMMY	019041
C		ARRAY 'T' DEFINED IN FINDT. IT IS	019042
C		THE MATRIX 'R' IN EQUATION III-23	019043
C		OF VOL 1 BUT NOT ORDERED EXACTLY	019044
C		AS SHOWN THEREIN	019045
C	LET:		019046
C	YY = THE NX BY 1 COLUMN MATRIX OF INDEPENDENT PLANT AND	CONTROL SYSTEM STATE VARIABLES (INCREASING INDEX	019047
C		ORDER)	019048
C	Y* = THE NX BY 1 COLUMN MATRIX OF INDEPENDENT PLANT AND	CONTROL SYSTEM STATE VARIABLES TO BE RETAINED (THE	019049
C		FIRST NX-NAUX ROWS) AND THE SENSOR AND TORQUE	019050
C		SIGNAL VARIABLES (LAST NAUX ROWS)	019051
C	R = THE NX BY NX SIMILARITY TRANSFORMATION MATRIX 'R'		019052
C	THEN		019053
C		YY(I) = R(I,J) * Y*(J)	019054
C	WHERE I=1 ... ,NX AND REPEATED INDEX J IMPLIES SUMMATION		019055
C			019056
C	' THE TRANSFORMED STATE	= NX BY 1 ARRAY COMPUTED IN FINDT AND	019057
C	VECTOR CORRELATION ARRAY	PRINTED THEREIN. IT IS STORED ONLY	019058
C	FOLLOWS' = IV	TEMPORARILY IN IV IN /LIJVV/. THE	019059
C		PRINTED ARRAY ELEMENTS HAVE THE	019060
C		FOLLOWING MEANING: ALL INDICES	019061
C		ARE CONSISTANT WITH THE LENDU AND	019062
C		LOCU ARRAYS	019063
C	(IV(I),I=1,NY2)	= INDICES ASSOCIATED WITH THE PLANT	019064
C		STATE VARIABLES WHICH ARE NOT	019065
C		ELIMINATED BY THE SIMILARITY	019066
C		TRANSFORMATION	019067
C	(IV(I),I=NY2+1,	= INDICES ASSOCIATED WITH THE SENSOR	019068
C	NY2+NXSS)	SIGNAL VARIABLES WHICH ARE TO BE	019069
C		USED FOR TRANSFER FUNCTION STUDIES	019070
C	(IV(I),I=NY2+NXSS+1,	= INDICES ASSOCIATED WITH THE CONTROL	019071
C	NX-NBTQ	SYSTEM STATE VARIABLES WHICH ARE	019072
C		NOT ELIMINATED BY THE SIMILARITY	019073
C		TRANSFORMATION	019074
C	(IV(I),I=NX-NBTQ+1,NX)	= INDICES ASSOCIATED WITH THE TORQUE	019075
C		SIGNAL VARIABLES WHICH ARE TO BE	019076
C		USED FOR TRANSFER FUNCTION STUDIES	019077
C	' OUTPUT MATRIX -I- '	= THE NX BY NX SIMILARITY TRANSFORMATION	019078
C		MATRIX RETURNED TO ASIMLR IN THE DUMMY	019079
C		ARRAY T FROM FINDT.	019080
C	' OUTPUT MATRIX Y*'	= THE NX BY 1 COLUMN MATRIX OF RETAINED	019081
C		INDEPENDENT SYSTEM STATE VARIABLES AND	019082
C		SENSOR AND TORQUE SIGNAL VARIABLES AS	019083
C			019084
C			019085

C		DEFINED ABOVE.	019086
C			019087
C	LET		019088
C		B = T**(-1) (MATRIX OPERATION)	019089
C	THEN		019090
C		Y*(I) = B(I,J)*YY(J)	019091
C			019092
C		WHERE YY IS DEFINED ABOVE AND SUMMATION IMPLIED BY REPEATED J	019093
C			019094
C	'R**(-1)*H*R MATRIX IS'	= THE NX BY NX MATRIX WHICH IS THE	019095
C		TRANSFORM OF THE PARTIAL DERIVATIVE	019096
C		MATRIX 'H' COMPUTED IN LINEAR, BY THE	019097
C		SIMILARITY TRANSFORMATION MATRIX 'R'	019098
C		COMPUTED IN FINDT. THIS IS THE MATRIX	019099
C		R**(-1)*H*R APPEARING IN EQUATION	019100
C		III-24 OF VOL 1; THE ROWS AND COLUMNS	019101
C		HOWEVER AT THIS POINT ARE CONSISTANT	019102
C		WITH Y* COLUMN MATRIX DEFINED ABOVE	019103
C	'RECRDERED R**(-1)*H*R	= THE NX BY NX MATRIX R**(-1)*H*R WHICH	019104
C	MATRIX IS'	HAS HAD ITS ROWS AND COLUMNS	019105
C		INTECHANGED SO AS TO BE CONSISTANT	019106
C		WITH THE TRANSFORMED STATE VECTOR	019107
C		CORRELATION ARRAY PRINTED AND DEFINED	019108
C		ABOVE. THIS IS THE MATRIX WHICH IS	019109
C		RETURNED TO DYN540. IT IS THE	019110
C		DESIRED SIMILARITY TRANSFORMATION	019111
C		MATRIX (NOT TO BE CONFUSED WITH THE	019112
C		MATRIX R ALONE)	019113
C			019114
C			019115
C	FREQUENCY DOMAIN ANALYSIS		019116
C	-----		019117
C			019118
C	DESCRIPTION OF TRANSFER FUNCTION	(THE PARAMETERS WHICH WILL	019119
C	-----	DEFINE WHAT TRANSFER	019120
C		FUNCTIONS ARE TO BE SETUP	019121
C		AND HOW THE FREQUENCY	019122
C		RESPONSE DATA IS TO BE	019123
C		DISPLAYED MUST BE USER	019124
C		SPECIFIED ON INPUT CARDS	019125
C		TO BE READ IN DYN540)	019126
C	LNAM = INPUT PARAMETER, STORED LOCALLY		019127
C	IF(LNAM.EQ.4+HTIME) LINEAR TIME RESPONSE ANALYSIS		019128
C	REQUESTED BY USER		019129
C	IF(LNAM.EQ.4+HFREQ) FREQUENCY ANALYSIS REQUESTED BY USER		019130
C	IF(LNAM.EQ.4H) FURTHER ANALYSIS OF LINEARIZED		019131
C	EQUATIONS NOT REQUESTED		019132
C	NCYC = TOTAL NUMBER OF TRANSFER FUNCTIONS WHICH WILL BE SETUP		019133
C	AND ANALYSED IN THE FREQUENCY DOMAIN, INPUT PARAMETER		019134
C	STORED LOCALLY, (MAXIMUM PER RUN EQUALS L1)		019135
C	LRV = TRANSFER FUNCTION DATA SPECIFICATION ARRAY, STORED		019136
C	LOCALLY AND DIMENSIONED (9,L1) TO AVOID EXCESSIVE USE		019137
C	OF SUBSCRIPTED VARIABLES THE FOLLOWING PARAMETERS ARE		019138
C	SET, FOR TRANSFER FUNCTION ICYC,ICYC=1,2,....NCYC		019139
C	ITYPE = LRV(1,ICYC) = TRANSFER FUNCTION TYPE, SEE SUBROUTINE		019140
C	ITYPE FOR DESCRIPTION OF 8 POSSIBLE TYPES AVAILABLE		019141
C	FOR USER		019142
C	ITHIN = LRV(2,ICYC) = INTEGER USED TO SPECIFY THE INPUT		019143
C	SIGNAL ASSOCIATED WITH THE TRANSFER FUNCTION		019144
C	JTFOUT = LRV(3,ICYC) = INTEGER USED TO SPECIFY THE OUTPUT		019145

```

C          SIGNAL ASSOCIATED WITH THE TRANSFER FUNCTION           019146
C KFLUT = LRY(4,ICYC) = INTEGER USED TO DEFINE IF PLOTS TO BE   019147
C          GENERATED                                              019148
C IAFLG = LRY(5,ICYC) = INTEGER USED TO DEFINE IF CHARACTERISTIC 019149
C          MATRIX OR ITS TRANSPOSE TO BE USED IN ANALYSIS         019150
C NEKP = LRY(6,ICYC) IF ITYPE.NE.7 UNUSED OTHERWISE = NUMBER OF 019151
C          FEEDBACK SIGNALS TO RETAIN (MAX OF 3 ALLOWED)          019152
C KEKP(I)=LRY(6+I,ICYC), I=1,NBKP = INTEGERS DEFINING WHICH    019153
C          FEEDBACK SIGNALS ARE TO BE RETAINED                   019154
C IRY      = TRANSFER FUNCTION COMPUTATIONAL TOLERANCE DATA ARRAY, 019155
C          STORED LOCALLY AND DIMENSIONED (3,L1). FOR SOME SPECIAL 019156
C          PROBLEMS DEFAULT TOLERANCE DATA WILL NOT WORK. THE DATA 019157
C          IN THIS ARRAY ALLOWS FOR OVERRIDE CAPABILITY            019158
C TCN     = TOLD = 10.00 ** IRY(1,ICYC) = ROOT TOLERANCE FACTOR 019159
C          ALL NUMERATOR AND DENOMINATOR ROOTS LESS THAN THIS FACTOR 019160
C          SET EQUAL TO ZERO IN SIFT. REAL*8 CONSTANT STORED IN   019161
C          /LTOL/                                                  019162
C GTOL    = 10.00 ** IRY(2,ICYC) = THE NUMERATOR GAIN MUST BE   019163
C          GREAT THAN GTOL FOR COMPUTATION TO CONTINUE            019164
C PTCL    = 10.00 ** IRY(3,ICYC) =ALL ROOTS OF SECOND TERM IN 019165
C          NUMERATOR OF TRANSFER FUNCTION LESS THAN PTOL SET     019166
C          EQUAL TO ZERO (SEE NUMS AND EQ III-42). THIS REFERS    019167
C          TO THE SPECIAL TECHNIQUE USED IN OBTAINING NUMERATOR   019168
C          ROOTS                                                    019169
C                                                         019170
C                                                         019171
C                                                         019172
C          ITYPE = MATRICES FOR TRANSFER FUNCTION COMPUTATION    019173
C          -----                                                019174
C DISCOS HAS THE BUILT IN CAPABILITY TO GENERATE UPON USER SELECTION 019176
C          ANY ONE OF EIGHT DIFFERENT TRANSFER FUNCTIONS. THE BASIC 019177
C          MODEL IN COMPUTER MEMORY WHEN ITYPE IS CALLED IS ILLUSTRATED 019178
C          BY THE FOLLOWING FIGURE                                  019179
C                                                         019180
C          +-----+                                             019181
C          | RT(S) + |----->| PLANT G(S) |----->0-----> 019182
C          | -       |-----+-----+-----> 019183
C          | FCS(S) |-----+-----+-----> 019184
C          |         |-----+-----+-----> 019185
C          |         |-----+-----+-----> 019186
C          |         |-----+-----+-----> 019187
C          |         |-----+-----+-----> 019188
C          |         |-----+-----+-----> 019189
C          |         |-----+-----+-----> 019190
C          |         |-----+-----+-----> 019191
C          |         |-----+-----+-----> 019192
C          |         |-----+-----+-----> 019193
C          |         |-----+-----+-----> 019194
C          |         |-----+-----+-----> 019195
C          |         |-----+-----+-----> 019196
C          |         |-----+-----+-----> 019197
C          |         |-----+-----+-----> 019198
C          |         |-----+-----+-----> 019199
C          |         |-----+-----+-----> 019200
C          |         |-----+-----+-----> 019201
C          |         |-----+-----+-----> 019202
C          |         |-----+-----+-----> 019203
C          |         |-----+-----+-----> 019204
C          |         |-----+-----+-----> 019205
C          |         |-----+-----+-----> 019206
C          |         |-----+-----+-----> 019207
C          |         |-----+-----+-----> 019208
C          |         |-----+-----+-----> 019209
C          |         |-----+-----+-----> 019210
C          |         |-----+-----+-----> 019211
C          |         |-----+-----+-----> 019212
C          |         |-----+-----+-----> 019213
C          |         |-----+-----+-----> 019214
C          |         |-----+-----+-----> 019215
C          |         |-----+-----+-----> 019216
C          |         |-----+-----+-----> 019217
C          |         |-----+-----+-----> 019218
C          |         |-----+-----+-----> 019219
C          |         |-----+-----+-----> 019220
C          |         |-----+-----+-----> 019221
C          |         |-----+-----+-----> 019222
C          |         |-----+-----+-----> 019223
C          |         |-----+-----+-----> 019224
C          |         |-----+-----+-----> 019225
C          |         |-----+-----+-----> 019226
C          |         |-----+-----+-----> 019227
C          |         |-----+-----+-----> 019228
C          |         |-----+-----+-----> 019229
C          |         |-----+-----+-----> 019230
C          |         |-----+-----+-----> 019231
C          |         |-----+-----+-----> 019232
C          |         |-----+-----+-----> 019233
C          |         |-----+-----+-----> 019234
C          |         |-----+-----+-----> 019235
C          |         |-----+-----+-----> 019236
C          |         |-----+-----+-----> 019237
C          |         |-----+-----+-----> 019238
C          |         |-----+-----+-----> 019239
C          |         |-----+-----+-----> 019240
C          |         |-----+-----+-----> 019241
C          |         |-----+-----+-----> 019242
C          |         |-----+-----+-----> 019243
C          |         |-----+-----+-----> 019244
C          |         |-----+-----+-----> 019245
C          |         |-----+-----+-----> 019246
C          |         |-----+-----+-----> 019247
C          |         |-----+-----+-----> 019248
C          |         |-----+-----+-----> 019249
C          |         |-----+-----+-----> 019250
C          |         |-----+-----+-----> 019251
C          |         |-----+-----+-----> 019252
C          |         |-----+-----+-----> 019253
C          |         |-----+-----+-----> 019254
C          |         |-----+-----+-----> 019255
C          |         |-----+-----+-----> 019256
C          |         |-----+-----+-----> 019257
C          |         |-----+-----+-----> 019258
C          |         |-----+-----+-----> 019259
C          |         |-----+-----+-----> 019260
C          |         |-----+-----+-----> 019261
C          |         |-----+-----+-----> 019262
C          |         |-----+-----+-----> 019263
C          |         |-----+-----+-----> 019264
C          |         |-----+-----+-----> 019265
C          |         |-----+-----+-----> 019266
C          |         |-----+-----+-----> 019267
C          |         |-----+-----+-----> 019268
C          |         |-----+-----+-----> 019269
C          |         |-----+-----+-----> 019270
C          |         |-----+-----+-----> 019271
C          |         |-----+-----+-----> 019272
C          |         |-----+-----+-----> 019273
C          |         |-----+-----+-----> 019274
C          |         |-----+-----+-----> 019275
C          |         |-----+-----+-----> 019276
C          |         |-----+-----+-----> 019277
C          |         |-----+-----+-----> 019278
C          |         |-----+-----+-----> 019279
C          |         |-----+-----+-----> 019280
C          |         |-----+-----+-----> 019281
C          |         |-----+-----+-----> 019282
C          |         |-----+-----+-----> 019283
C          |         |-----+-----+-----> 019284
C          |         |-----+-----+-----> 019285
C          |         |-----+-----+-----> 019286
C          |         |-----+-----+-----> 019287
C          |         |-----+-----+-----> 019288
C          |         |-----+-----+-----> 019289
C          |         |-----+-----+-----> 019290
C          |         |-----+-----+-----> 019291
C          |         |-----+-----+-----> 019292
C          |         |-----+-----+-----> 019293
C          |         |-----+-----+-----> 019294
C          |         |-----+-----+-----> 019295
C          |         |-----+-----+-----> 019296
C          |         |-----+-----+-----> 019297
C          |         |-----+-----+-----> 019298
C          |         |-----+-----+-----> 019299
C          |         |-----+-----+-----> 019300
C          |         |-----+-----+-----> 019301
C          |         |-----+-----+-----> 019302
C          |         |-----+-----+-----> 019303
C          |         |-----+-----+-----> 019304
C          |         |-----+-----+-----> 019305
C          |         |-----+-----+-----> 019306
C          |         |-----+-----+-----> 019307
C          |         |-----+-----+-----> 019308
C          |         |-----+-----+-----> 019309
C          |         |-----+-----+-----> 019310
C          |         |-----+-----+-----> 019311
C          |         |-----+-----+-----> 019312
C          |         |-----+-----+-----> 019313
C          |         |-----+-----+-----> 019314
C          |         |-----+-----+-----> 019315
C          |         |-----+-----+-----> 019316
C          |         |-----+-----+-----> 019317
C          |         |-----+-----+-----> 019318
C          |         |-----+-----+-----> 019319
C          |         |-----+-----+-----> 019320
C          |         |-----+-----+-----> 019321
C          |         |-----+-----+-----> 019322
C          |         |-----+-----+-----> 019323
C          |         |-----+-----+-----> 019324
C          |         |-----+-----+-----> 019325
C          |         |-----+-----+-----> 019326
C          |         |-----+-----+-----> 019327
C          |         |-----+-----+-----> 019328
C          |         |-----+-----+-----> 019329
C          |         |-----+-----+-----> 019330
C          |         |-----+-----+-----> 019331
C          |         |-----+-----+-----> 019332
C          |         |-----+-----+-----> 019333
C          |         |-----+-----+-----> 019334
C          |         |-----+-----+-----> 019335
C          |         |-----+-----+-----> 019336
C          |         |-----+-----+-----> 0193
```

```

C          X(I,S) = Y(I,S) = I-TH INDEPENDENT PERTURBATION VARIABLE 019206
C          FOR THE PLANT WHICH HAS NOT BEEN 019207
C          EXCHANGED BY THE SIMILARITY 019208
C          TRANSFORMATION FOR A SENSOR SIGNAL 019209
C      FOR INDEX I; I=NY2+1,...,NY2+NXSS 019210
C          X(I,S) = XSS(L,S) = L-TH SENSOR SIGNAL (L=I-NY2) WHICH BY 019211
C          THE SIMILARITY TRANSFORMATION IS USED 019212
C          AS A PLANT STATE VARIABLE 019213
C      FOR INDEX I; I=NY2+NXSS+1,...,NX-NBTQ 019214
C          X(I,S) = Y(M,S) = M-TH INDEPENDENT PERTURBATION VARIABLE 019215
C          FOR THE CONTROL SYSTEM (M=I-NY2-NXSS) 019216
C          WHICH HAS NOT BEEN EXCHANGED BY THE 019217
C          SIMILARITY TRANSFORMATION FOR A TORQUE 019218
C          SIGNAL 019219
C      FOR INDEX I; I=NX-NBTQ+1,...,NX 019220
C          X(I,S) = TQS(N,S) = N-TH TORQUE SIGNAL (N=I-NX+NBTQ) WHICH 019221
C          BY THE SIMILARITY TRANSFORMATION IS 019222
C          USED AS A CONTROL SYSTEM STATE 019223
C          VARIABLE 019224
C      FOR INDICES I AND J; I=1,...,NX AND J=1,...,NX 019225
C          A(I,J) = R*(-1)*H*K(I,J) = (I,J) ELEMENT OF THE 019226
C          SIMILARITY TRANSFORMATION MATRIX 019227
C          RETURNED TO DYN540 FROM ASIMLR 019228
C      FOR INDEX K; K=1,NBTQ 019229
C          U(K,S) = RT(K,S) = REFERENCE INPUT TORQUE SIGNAL THAT THE 019230
C          K-TH CONTROL TORQUE SIGNAL TQS(K,S) IS 019231
C          SUBTRACTED FROM AND FEED TO THE PLANT 019232
C      FOR INDEX K; K=NBTQ+1,...,NBTQ+NXSS 019233
C          U(K,S) = RS(L,S) = REFERENCE INPUT SENSOR SIGNAL WHICH IS 019234
C          ADDED TO THE L-TH SENSOR SIGNAL 019235
C          XSS(L,S) AND FEED INTO THE CONTROL 019236
C          SYSTEM (L=K-NBTQ) 019237
C      FOR INDEX I; I=1,...,NY2+NXSS AND K; K=1,...,NBTQ 019238
C          B(I,K) = -A(I,K) = (I,K) ELEMENT OF THE REFERENCE INPUT 019239
C          COEFFICIENT MATRIX, THESE ELEMENTS ARE 019240
C          ASSOCIATED WITH REFERENCE INPUT TORQUE 019241
C          SIGNALS FEED INTO THE PLANT 019242
C      FOR INDEX I; I=1,...,NY2+NXSS AND K; K=NBTQ+1,...,NBTQ+NXSS 019243
C          E(I,K) = 0 019244
C      FOR INDEX I; I=NY2+NXSS+1,...,NX AND K; K=1,...,NBTQ 019245
C          E(I,K) = 0 019246
C      FOR INDEX I; I=NY2+NXSS+1,...,NX AND K; K=NBTQ+1,...,NBTQ+NXSS 019247
C          B(I,K) = A(I,K) = (I,K) ELEMENT OF THE REFERENCE INPUT 019248
C          COEFFICIENT MATRIX, THESE ELEMENTS ARE 019249
C          ASSOCIATED WITH REFERENCE INPUT SENSOR 019250
C          SIGNALS FEED INTO THE CONTROL 019251
C 019252
C      IN GENERAL THE ELEMENTS X(I,S), I=1,...,NX DEFINE ALL POSSIBLE 019253
C      OUTPUT SIGNALS AND THE ELEMENTS U(K,S), K=1,...,NAUX DEFINE 019254
C      ALL POSSIBLE INPUT SENSOR AND TORQUE REFERENCE SIGNALS. IN 019255
C      THE S-DOMAIN THE EQUATIONS MAY BE EXPRESSED AS 019256
C 019257
C          (I(I,J)*S - A(I,J))*X(J,S) = B(I,K)*U(K,S) 019258
C 019259
C      WHERE 019260
C          I(I,J) = 1 IF (I.EQ.J) 019261
C          I(I,J) = 0 IF (I.NE.J) 019262
C          I = 1,...,NX 019263
C          J = 1,...,NX 019264
C          K = 1,...,NAUX 019265

```

```

C
C
C THE COEFFICIENT MATRIX 'A' COMPLETELY DEFINES ALL COUPLING BETWEEN 019266
C THE NX LINEARLY INDEPENDENT STATE VARIABLES, SENSOR SIGNALS 019267
C AND TORQUE SIGNALS USED TO MODEL THE DYNAMIC RESPONSE OF THE 019268
C SYSTEM 019269
C 019270
C 019271
C 019272
C SEVERAL DIFFERENT TYPES OF TRANSFER FUNCTIONS ARE USED IN THE 019273
C LINEAR STABILITY ANALYSIS OF A PARTICULAR SYSTEM. THE 'TYPES 019274
C OF TRANSFER FUNCTIONS' MAY BE CLASSIFIED ACCORDING TO WHAT 019275
C FEEDBACK INFORMATION IS ALLOWED; THAT IS, WHICH FEEDBACK LOOPS 019276
C ARE OPEN AND WHICH FEEDBACK LOOPS ARE CLOSED. 019277
C 019278
C FEEDBACK LOOPS ARE CUT IN THE DISCOS FORMULATION BY SELECTIVELY 019279
C SETTING CERTAIN ELEMENTS IN THE 'A' MATRIX EQUAL TO ZERO TO 019280
C FORM THE REDUCED MATRIX 'AR'. THE ASSOCIATED TRANSFER FUNCTION 019281
C WHICH DEFINES THE OUTPUT RESPONSE OF THE P-TH STATE VARIABLE 019282
C X(P,S) TO THE Q-TH REFERENCE INPUT SIGNAL U(Q,S) IS FROM 019283
C CRAMER'S RULE DEFINED BY 019284
C 019285
C 
$$\frac{X(P,S)}{U(Q,S)} = \frac{\text{DET}(AUG(I*S - AR))}{\text{DET}(I*S - AR)}$$

C 019286
C 019287
C 019288
C 019289
C WHERE 019290
C I = UNIT DIAGONAL MATRIX 019291
C AUG(I*S - AR) = AUGMENTED MATRIX OBTAINED BY PLACING COLUMN 019292
C Q OF THE 'B' MATRIX (REFERENCE INPUT SIGNAL 019293
C COEFFICIENT MATRIX) INTO COLUMN P OF THE 019294
C '(I*S - AR)' MATRIX 019295
C DET = 'DETERMINANT OF' 019296
C 019297
C SUBROUTINE Tftype ACCEPTS FROM DYN540 THE COEFFICIENT MATRIX 'A' 019298
C AND INTEGER CODES WHICH DEFINE WHICH FEEDBACK LOOPS ARE TO BE 019299
C CUT AND WITH RESPECT TO WHICH REFERENCE INPUT SIGNAL THE 019300
C TRANSFER FUNCTION IS TO BE OBTAINED. IT RETURNS TO DYN540 019301
C THE REDUCED MATRIX 'AR', ITS DIMENSION AND THE COLUMN OF THE 019302
C 'B' MATRIX ASSOCIATED WITH THE REFERENCE INPUT SIGNAL OF 019303
C INTEREST 019304
C 019305
C ITYPE = INTEGER CODE USED TO DEFINE THE 'TYPE' OF TRANSFER 019306
C FUNCTION' TO BE GENERATED 019307
C JCOL = INTEGER CODE USED TO DEFINE PARTICULAR REFERENCE INPUT 019308
C SIGNAL OF INTEREST 019309
C A = DUMMY ARRAY IN WHICH THE 'A' MATRIX IS PASSED TO Tftype 019310
C Z = DUMMY ARRAY IN WHICH THE 'AR' MATRIX IS RETURNED 019311
C E = DUMMY ARRAY IN WHICH THE COLUMN OF THE 'B' MATRIX 019312
C ASSOCIATED WITH THE REFERENCE INPUT SIGNAL OF INTEREST 019313
C IS RETURNED 019314
C NZ = INTEGER, DIMENSION OF THE 'AR' MATRIX 019315
C NOKP = SPECIAL FOR ITYPE=7 019316
C KOKP = SPECIAL FOR ITYPE=7 019317
C 019318
C 019319
C -- TRANSFER FUNCTION SETUP -- 019320
C 019321
C ITYPE = 1 = FORWARD PATH TRANSFER FUNCTION G(S). THE 019322
C REDUCED MATRIX 'AR' IS (NY2+NXSS) BY (NY2+NXSS) 019323
C THE REFERENCE INPUT TORQUE SIGNAL OF INTEREST IS 019324
C TORQUE SIGNAL JCOL WHERE 0 < JCOL < NBTO+1. 019325

```

C	THE FOLLOWING CODE IS USED IN IFTYPE	019326
C	NZ = NY2+NXSS	019327
C	JB = NY2+NXSS+ND2+JCOL	019328
C	DO 1 I=1,NZ	019329
C	B(I) = A(I,JB)	019330
C	DO 1 J=1,NZ	019331
C	Z(I,J) = A(I,J)	019332
C	1 CONTINUE	019333
C	IF(IITYPE.EQ.-1) THE SIGNS OF THE ELEMENTS IN 'B'	019334
C	ARE ALL CHANGED (NEGATIVE FEEDBACK FOR NUMERATOR)	019335
C	THE TRANSFER FUNCTION BETWEEN ANY OUTPUT SENSOR	019336
C	SIGNAL AND REFERENCE INPUT TORQUE SIGNAL JCCL	019337
C	MAY BE OBTAINED	019338
C		019339
C	IITYPE = 2 = FEEDBACK PATH TRANSFER FUNCTION H(S). THE	019340
C	REDUCED MATRIX 'AR' IS (ND2+NBTQ) BY (ND2+NBTQ)	019341
C	THE REFERENCE INPUT SENSOR SIGNAL OF INTEREST IS	019342
C	SENSOR SIGNAL JCOL WHERE 0 < JCOL < NXSS+1	019343
C	THE FOLLOWING CODE IS USER IN IFTYPE	019344
C	NZ = ND2+NBTQ	019345
C	JB = NY2+JCOL	019346
C	DO 1 I=1,NZ	019347
C	II = NY2+NXSS+1	019348
C	B(I) = A(II,JB)	019349
C	DO 1 J=1,NZ	019350
C	JJ = NY2+NXSS+J	019351
C	Z(I,J) = A(II,JJ)	019352
C	1 CONTINUE	019353
C	THE TRANSFER FUNCTION BETWEEN ANY OUTPUT TORQUE	019354
C	SIGNAL AND REFERENCE INPUT SENSOR SIGNAL JCCL	019355
C	MAY BE OBTAINED	019356
C		019357
C	IITYPE = 3 = OPEN LOOP TRANSFER FUNCTION G(S)*H(S). THE	019358
C	FEEDBACK PATH FROM THE CONTROL SYSTEM TO THE	019359
C	PLANT IS COMPLETELY CUT. THIS IS EFFECTED BY	019360
C	SETTING ALL COEFFICIENTS ASSOCIATED WITH CONTROL	019361
C	SYSTEM STATE VARIABLES, IN THE EQUATIONS FOR THE	019362
C	PLANT STATE VARIABLES, EQUAL TO ZERO. THE REDUCED	019363
C	MATRIX 'AR' IS NX BY NX AND THE REFERENCE INPUT	019364
C	TORQUE SIGNAL OF INTEREST IS TORQUE SIGNAL JCOL	019365
C	WHERE 0 < JCOL < NBTQ+1. THE FOLLOWING CODE IS	019366
C	USED IN IFTYPE	019367
C	NZ = NX	019368
C	NN = NY2+NXSS	019369
C	JB = NY2+NXSS+ND2+JCOL	019370
C	DO 1 I=1,NX	019371
C	B(I) = 0.000	019372
C	IF(1.LE.NN) B(I) = A(I,JB)	019373
C	DO 1 J=1,NX	019374
C	Z(I,J) = 0.000	019375
C	IF(1.LE.NN.AND.J.GT.NN) GO TO 1	019376
C	Z(I,J) = A(I,J)	019377
C	1 CONTINUE	019378
C	IF(IITYPE.EQ.-3) THE SIGNS OF THE ELEMENTS IN 'B'	019379
C	ARE ALL CHANGED. THE TRANSFER FUNCTION BETWEEN	019380
C	ANY OUTPUT TORQUE SIGNAL AND REFERENCE INPUT	019381
C	TORQUE SIGNAL JCOL MAY BE OBTAINED	019382
C		019383
C	IITYPE = 4 = OPEN LOOP TRANSFER FUNCTION H(S)*G(S). THE	019384
C	FEEDBACK PATH FROM THE PLANT TO THE CONTROL	019385

C	SYSTEM IS COMPLETELY CUT. THIS IS EFFECTED BY	019386
C	SETTING ALL COEFFICIENTS ASSOCIATED WITH PLANT	019387
C	STATE VARIABLES, IN THE EQUATIONS FOR THE CONTROL	019388
C	STATE VARIABLES, EQUAL TO ZERO. THE REDUCED	019389
C	MATRIX 'AR' IS NX BY NX AND THE REFERENCE INPUT	019390
C	SENSOR SIGNAL OF INTEREST IS JCOL. HERE	019391
C	$0 < JCOL < NXSS+1$. THE FOLLOWING CODE IS USED	019392
C	IN TTYPE	019393
C	NZ = NX	019394
C	NN = NY2+NXSS	019395
C	JB = NY2+JCOL	019396
C	DO 1 I=1,NX	019397
C	B(I) = 0.000	019398
C	IF(I.GT.NN) B(I) = A(I,JB)	019399
C	DO 1 J=1,NX	019400
C	Z(I,J) = 0.000	019401
C	IF(I.GT.NN.AND.J.LE.NN) GO TO 1	019402
C	Z(I,J) = A(I,J)	019403
C	1 CONTINUE	019404
C	THE TRANSFER FUNCTION BETWEEN ANY OUTPUT SENSOR	019405
C	SIGNAL AND REFERENCE INPUT SENSOR SIGNAL JCOL	019406
C	MAY BE OBTAINED	019407
C		019408
C	ITYPE = 5 = CLOSED LOOP TRANSFER FUNCTION - CONTROL RATIO	019409
C	$G(S)/(1+G(S)*H(S))$. ALL FEEDBACK PATHS ARE CLOSED	019410
C	AND HENCE THE REDUCED MATRIX 'AR' IS NX BY NX AND	019411
C	EQUAL TO THE INPUTTED 'A' MATRIX. THE REFERENCE	019412
C	INPUT TORQUE SIGNAL OF INTEREST IS JCOL WHERE	019413
C	$0 < JCOL < NBTC+1$. THE FOLLOWING CODE IS USED	019414
C	IN TTYPE	019415
C	NZ = NX	019416
C	NN = NY2+NXSS	019417
C	JB = NY2+NXSS+ND2+JCOL	019418
C	DO 1 I=1,NX	019419
C	B(I) = 0.000	019420
C	IF(I.LE.NN) B(I) = A(I,JB)	019421
C	DO 1 J=1,NX	019422
C	Z(I,J) = A(I,J)	019423
C	1 CONTINUE	019424
C	IF(ITYPE.EQ.-5) THE SIGNS OF THE ELEMENTS IN 'B'	019425
C	ARE ALL CHANGED. THE TRANSFER FUNCTION BETWEEN	019426
C	ANY OUTPUT SENSOR SIGNAL AND THE REFERENCE INPUT	019427
C	TORQUE SIGNAL JCOL MAY BE OBTAINED	019428
C		019429
C	ITYPE = 6 = CLOSED LOOP TRANSFER FUNCTION	019430
C	$G(S)*H(S)/(1+G(S)*H(S))$. ALL FEEDBACK PATHS ARE	019431
C	CLOSED AND HENCE THE REDUCED MATRIX 'AR' IS	019432
C	NX BY NX AND EQUAL TO THE INPUTTED 'A' MATRIX.	019433
C	THE REFERENCE INPUT SENSOR SIGNAL OF INTEREST IS	019434
C	JCOL WHERE $0 < JCOL < NXSS+1$. THE FOLLOWING CODE	019435
C	IS USED IN TTYPE	019436
C	NZ = NX	019437
C	NN = NY2+NXSS	019438
C	JB = NY2+JCOL	019439
C	DO 1 I=1,NX	019440
C	B(I) = 0.000	019441
C	IF(I.GT.NN) E(I) = A(I,JB)	019442
C	DO 1 J=1,NX	019443
C	Z(I,J) = A(I,J)	019444
C	1 CONTINUE	019445


```

C      THE TRANSFER FUNCTION BETWEEN ANY OUTPUT SENSOR 019446
C      SIGNAL AND THE REFERENCE INPUT SENSOR SIGNAL 019447
C      JCUL MAY BE OBTAINED 019448
C 019449
C      ITYPE = 7 = QUASI OPEN LOOP TRANSFER FUNCTION. THE FEEDBACK 019450
C      PATH FROM THE CONTROL SYSTEM TO THE PLANT IS 019451
C      PARTIALLY CLOSED. THE NBKP TORQUE SIGNALS 019452
C      KBKP(1),...,KBKP(NBKP) ARE THE ONLY TORQUE 019453
C      SIGNALS FEEDBACK FROM THE CONTROL SYSTEM TO THE 019454
C      PLANT. ALL OTHER LOOPS ARE OPEN (NBKP.LE.3). THIS 019455
C      IS EFFECTED BY SETTING ALL COEFFICIENTS 019456
C      ASSOCIATED WITH THE CONTROL SYSTEM STATE 019457
C      VARIABLES, IN THE EQUATIONS FOR THE PLANT STATE 019458
C      VARIABLES, EQUAL TO ZERO EXCEPT THOSE ASSOCIATED 019459
C      WITH THE NBKP TORQUE SIGNALS TO BE FEEDBACK. THE 019460
C      REDUCED MATRIX 'A*' IS NX BY NX AND THE REFERENCE 019461
C      INPUT TORQUE SIGNAL OF INTEREST IS TORQUE SIGNAL 019462
C      JCUL WHERE 0 < JCUL < NXSS+1. THE FOLLOWING CODE 019463
C      IS USED IN ITYPE 019464
C      NZ = NX 019465
C      NN = NY2+NXSS 019466
C      JB = NY2+NXSS+ND2+JCUL 019467
C      DO 1 I=1,NX 019468
C      B(I) = 0.000 019469
C      IF(I.LE.NN) B(I) = A(I,JB) 019470
C      DO 1 J=1,NX 019471
C      Z(I,J) = A(I,J) 019472
C      IF(J.LE.NN.OR.I.EQ.NN) GO TO 1 019473
C      Z(I,J) = 0.000 019474
C      DO 2 K=1,NBTQ 019475
C      LB = NY2+NXSS+ND2+KBTQ(K) 019476
C      IF(J.EQ.LB) Z(I,J) = A(I,LB) 019477
C      2 CONTINUE 019478
C      1 CONTINUE 019479
C      IF(ITYPE.EQ.-7) THE SIGNS OF THE ELEMENTS IN 'B' 019480
C      ARE ALL CHANGED. THE TRANSFER FUNCTION BETWEEN 019481
C      ANY OUTPUT TORQUE SIGNAL AND THE REFERENCE INPUT 019482
C      TORQUE SIGNAL JCUL MAY BE OBTAINED 019483
C 019484
C      ***** SPECIAL CASE ***** 019485
C 019486
C      A REFERENCE INPUT SIGNAL ASSOCIATED WITH ANY SYSTEM STATE 019487
C      VARIABLE X(I,S), I=1,2,...,NX MAY BE FEED INTO THE SYSTEM. 019488
C      EITHER ALL STATE VARIABLES OR ALL EXCEPT THE ONE ASSOCIATED 019489
C      WITH THE REFERENCE INPUT SIGNAL MAY BE FEEDBACK TO THE SYSTEM. 019490
C      EITHER THE CLOSED LOOP OR THE ONE STATE OPEN LOOP TRANSFER 019491
C      FUNCTION MAY BE OBTAINED BETWEEN ANY STATE VARIABLE AND THE 019492
C      REFERENCE INPUT SIGNAL 019493
C 019494
C      ITYPE = 8 = ONE STATE OPEN LOOP TRANSFER FUNCTION. 019495
C      A REFERENCE INPUT SIGNAL MAY BE SUMMED WITH ANY 019496
C      SYSTEM STATE VARIABLE X(JCUL,S), 0 < JCUL < NX+1 019497
C      AND FEED INTO THE TOTAL SYSTEM. FOR THIS CASE 019498
C      THE LOOP WHICH DEFINES X(JCUL,S) FEEDBACK INTO 019499
C      THE SYSTEM IS CUT. THIS IS EFFECTED BY SETTING 019500
C      TO ZERO ALL STATE VARIABLE COUPLING COEFFICIENTS 019501
C      ASSOCIATED WITH STATE VARIABLE JCUL FEEDBACK. THE 019502
C      FOLLOWING CODE IS USED IN ITYPE 019503
C      NZ = NX 019504
C      JB = JCUL 019505

```

```

C          DO 1 I=1,NX                                019506
C          B(I) = 0.000                                019507
C          IF (I.NE.JB) B(I) = -A(I,JB)               019508
C          DO 1 J=1,NX                                019509
C          Z(I,J) = A(I,J)                            019510
C          IF (J.NE.JB) GO TO 1                        019511
C          IF (I.EQ.JB) GO TO 1                        019512
C          Z(I,J) = 0.000                              019513
C          1 CONTINUE                                  019514
C          THE TRANSFER FUNCTION BETWEEN ANY STATE VARIABLE 019515
C          AND ANY REFERENCE INPUT SIGNAL MAY BE OBTAINED 019516
C          (NEGATIVE FEEDBACK USED)                    019517
C
C          ITYPE = -8 = CLOSED LOOP TRANSFER FUNCTION. A REFERENCE INPUT 019518
C          SIGNAL MAY BE SUMMED WITH ANY SYSTEM STATE      019519
C          VARIABLE X(JCOL,S), 0 < JCOL < NX+1 AND FEED INTO 019520
C          THE TOTAL SYSTEM. FOR THIS CASE ALL FEEDBACK   019521
C          LOOPS ARE CLOSED. THE FOLLOWING CODE IS USED   019522
C          IN IFTYPE                                     019523
C          NZ = NX                                       019524
C          JB = JCOL                                    019525
C          DO 1 I=1,NX                                  019526
C          B(I) = 0.000                                019527
C          IF (I.NE.JB) B(I) = -A(I,JCOL)               019528
C          DO 1 J=1,NX                                  019529
C          Z(I,J) = A(I,J)                             019530
C          1 CONTINUE                                  019531
C          THE TRANSFER FUNCTION BETWEEN ANY STATE VARIABLE 019532
C          AND ANY REFERENCE INPUT SIGNAL MAY BE OBTAINED 019533
C          (NEGATIVE FEEDBACK USED)                    019534
C
C          IF DEBUG(67) = .TRUE. THE FOLLOWING CAPTIONS AND MATRICES ARE 019535
C          PRINTED:                                     019536
C          * SUBROUTINE IFTYPE, TRANSFER = INPUT TRANSFER FUNCTION INTEGER 019537
C          FUNCTION TYPE'                               019538
C          * THE CHARACTERISTIC MATRIX = THE NZ BY NZ REDUCED MATRIX 'AR' 019539
C          IS'                                           019540
C          WHICH IS TO BE RETURNED TO DYN540            019541
C          IN THE DUMMY ARRAY Z                          019542
C          * THE CORRESPONDING INPUT = THE NZ BY 1 COLUMN MATRIX 'B'      019543
C          COEFFICIENTS ARE'                             019544
C          WHICH IS TO BE RETURNED TO DYN540            019545
C          IN THE DUMMY ARRAY B                          019546
C
C          THE TRANSFER FUNCTION IN FACTORED POLYNOMIAL FORM 019547
C          ----- 019548
C          SUBROUTINE IFTYPE ACCEPTS THE COEFFICIENT MATRIX 'A' OF THE 019549
C          LINEARIZED EQUATIONS OF MOTION. IT ALSO ACCEPTS THE USER 019550
C          DEFINED INTEGER CODES WHICH DEFINE WHAT TYPE OF TRANSFER 019551
C          FUNCTION IS DESIRED AND WHICH STATE VARIABLE WILL BE HAVE ITS 019552
C          FEEDBACK MODIFIED BY A REFERENCE INPUT SIGNAL. 019553
C          SUBROUTINE IFTYPE RETURNS TO DYN540 THE REDUCED MATRIX 'AR' WHICH 019554
C          IS THE CHARACTERISTIC MATRIX FOR THE TRANSFER FUNCTION OF 019555
C          INTEREST. IT ALSO RETURNS THE COLUMN MATRIX 'B' WHICH IS THE 019556
C          REFERENCE INPUT SIGNAL COEFFICIENT MATRIX, AND NZ ITS ORDER 019557
C          UPON RETURNING TO DYN540 FROM IFTYPE THE FOLLOWING CAPTIONS AND 019558
C          MATRICES WILL BE PRINTED:                    019559
C          * OUTPUT MATRIX -AR- (NZ X NZ) = CHARACTERISTIC MATRIX COMPUTED 019560
C

```

```

C                                     IN IFTYPE AND RETURNED TO DYN540 019566
C                                     IN DUMMY ARRAY #1, THIS IS THE 019567
C                                     REDUCED MATRIX 'AR' 019568
C      ' OUTPUT MATRIX BCUL ( 1 X NZ) = THE NZ BY 1 COLUMN MATRIX 'B' 019569
C                                     COMPUTED IN IFTYPE AND RETURNED 019570
C                                     TO DYN540 IN DUMMY ARRAY V1, THIS 019571
C                                     IS THE COLUMN MATRIX OF REFERENCE 019572
C                                     INPUT SIGNAL COEFFICIENTS TO BE 019573
C                                     USED FOR TRANSFER FUNCTION 019574
C                                     COMPUTATION 019575
C                                     019576
C      THESE MATRICES ALONG WITH OTHER STORED QUANTITIES PROVIDE 019577
C      SUFFICIENT DATA TO CONSTRUCT THE DESIRED TRANSFER FUNCTION VIA 019578
C      APPLICATION OF CRAMER'S RULE, THAT IS 019579
C                                     019580
C                                     
$$\frac{X(P,S)}{U(Q,S)} = \frac{\text{DET}(AUG(I*S - AR))}{\text{DET}(I*S - AR)}$$
 019581
C                                     019582
C                                     019583
C      WHERE THE DESIRED OUTPUT SIGNAL X(P,S) IS DETERMINED FROM THE 019584
C      INTEGER CODES JTFOUT AND ITYPE. 019585
C                                     019586
C                                     019587
C                                     019588
C      DENOMINATOR ROOTS 019589
C      ----- 019590
C                                     019591
C      THE FACTORED POLYNOMIAL FORM OF THE DENOMINATOR IS FOUND BY A 019592
C      SIMPLE CALL TO THE QR-ALGORITHM WITH THE MATRIX 'AR' TO FIND 019593
C      ITS COMPLEX ROOTS. THE USER MAY REQUEST THAT 'AR TRANSPOSE' 019594
C      BE USED INSTEAD OF 'AR', THIS AT TIMES FOR SPECIAL PROBLEMS 019595
C      CAN BE USEFUL IN OVERCOMING NUMERICAL PROBLEMS. IN EITHER CASE 019596
C      THE ROOTS OF 'AR' AND 'AR**T' ARE BOTH FOUND AND PRINTED UNDER 019597
C      THE FOLLOWING CAPTIONS: 019598
C      'R AR' = A LISTING IN DECREASING ORDER 019599
C      'NO REAL PART IMAGINARY PART' OF REAL PART OF ALL NZ COMPLEX 019600
C      'RART' = A LISTING IN DECREASING ORDER 019601
C      'NO REAL PART IMAGINARY PART' OF REAL PART OF ALL NZ COMPLEX 019602
C      ROOTS OF THE 'AR**T' MATRIX 019603
C      019604
C      ** NOTE ** 019605
C      BOTH SETS OF COMPLEX ROOTS 019606
C      SHOULD BE EQUAL. THIS IS 019607
C      A CHECK, THE ROOTS OF 'AR' 019608
C      WILL BE USED BY DEFAULT 019609
C                                     019610
C      LET: 019611
C      RRK(K) = REAL PART OF K-TH COMPLEX ROOT OF 'AR' 019612
C      RID(K) = IMAGINARY PART OF K-TH COMPLEX ROOT OF 'AR' 019613
C      THEN 019614
C      DET(I*S - AR) = (S - RRK( 1) - J*RID( 1)) 019615
C      * (S - RRK( 2) - J*RID( 2)) 019616
C      . 019617
C      . 019618
C      . 019619
C      . 019620
C      * (S - RRK(NZ) - J*RID(NZ)) 019621
C      WHERE 019622
C      J=SQRT(-1.0) 019623
C      AND REAL AND IMAGINARY PARTS .LE. TOLD ARE SET TO ZERO 019624
C                                     019625

```

```

C
C      NUMERATOR ROOTS
C      -----
C
C      THE FACTORED POLYNOMIAL FORM OF THE NUMERATOR IS NOT AS STRAIGHT-
C      FORWARD TO OBTAIN. THIS IS DONE IN THE SUBROUTINE NUMS
C      DISCUSSED BELOW. THE DIFFICULTY IN OBTAINING THE ROOTS OF
C      THE NUMERATOR COMES FROM THE FACT THAT ITS DEGREE MAY (AND
C      USUALLY IS) LESS THAN THAT OF THE DENOMINATOR.
C
C      NUMS RECEIVES FROM DYN540 THE REDUCED MATRIX 'AR', THE COLUMN
C      MATRIX 'B' AND THE COLUMN LOCATION P OF THE OUTPUT STATE
C      VARIABLE X(P,S). IT FORMS THE AUGMENTED MATRIX 'AUG(I*S - AR)'
C      BY REPLACING COLUMN P OF '(I*S - AR)' WITH THE 'B' COLUMN
C      MATRIX TO FORM 'AUG(I*S - AR)'. USING SPECIAL TECHNIQUES
C      THE DETERMINATE IS FORMED AND RETURNED TO DYN540 IN FACTORED
C      POLYNOMIAL FORM WHERE THE FOLLOWING CAPTIONS AND MATRICES
C      ARE PRINTED:
C      'NUM'      = A LISTING IN DECREASING ORDER
C      'NO REAL PART IMAGINARY PART' OF REAL PARTS OF ALL COMPLEX
C      ROOTS OF THE TRANSFER FUNCTION
C      'DEN'      = A LISTING IN DECREASING ORDER
C      'NO REAL PART IMAGINARY PART' OF REAL PARTS OF ALL COMPLEX
C      ROOTS OF THE TRANSFER FUNCTION
C      DENOMINATOR
C
C      ** NOTE **
C      IF THE NUMBER OF NUMERATOR
C      ROOTS NN IS LESS THAN THE
C      NUMBER OF DENOMINATOR ROOTS
C      NZ THEN UNDER THE TWO COLUMNS
C      WITH CAPTION HEADING 'NUM'
C      THE LAST (NZ-NN) ROWS WILL
C      BE BLANK
C
C      LET:
C      GAIN      = NUMERATOR ROOT GAIN
C      NN        = TOTAL NUMBER OF NUMERATOR ROOTS
C      RRN(K)    = REAL PART OF K-TH COMPLEX ROOT OF NUMERATOR
C      RIN(K)    = IMAGINARY PART OF K-TH COMPLEX ROOT OF NUMERATOR
C
C      THEN
C      DET(AUG(I*S - AR)) = GAIN*(S - RRN( 1) - J*RIN( 1))
C                          *(S - RRN( 2) - J*RIN( 2))
C                          .
C                          .
C                          .
C                          *(S - RRN(NN) - J*RIN(NN))
C
C      WHERE
C      J=SQRT(-1.0)
C      AND REAL AND IMAGINARY PARTS .LE. TOLN ARE SET TO ZERO
C
C      NUMS - FIND TRANSFER FUNCTION NUMERATOR ROOTS
C      -----
C
C      THE COMPUTATION OF THE ROOTS OF THE NUMERATOR PRESENTS A PROBLEM
C      WHICH MUST BE OVERCOME. THE ALGORITHM MUST BE ABLE TO DETERMINE
C      NOT ONLY THE DEGREE OF THE POLYNOMIAL BUT ALSO ITS ROOTS.
C      THE METHOD USED MAKES A QUICK CHECK TO SEE IF THE PIVOT
C      ELEMENT IS ZERO; IF NOT, BOTH NUMERATOR AND DENOMINATOR HAVE

```

C	THE SAME DEGREE AND NUMERATOR ROOTS ARE READILY OBTAINABLE	C19686
C	BY A DIRECT APPLICATION OF THE QR-ALGORITHM.	C19687
C	IF THE PIVOT ELEMENT IS ZERO THE DEGREE OF THE NUMERATOR IS	C19688
C	AT LEAST 1 LESS THAN THE DENOMINATOR. THE THEORY USED TO SOLVE	C19689
C	THIS PROBLEM IS ON PAGES 71-73, VOL 1.	C19690
C	EASILY, THE SUBSTITUTION	C19691
C		C19692
C	$(S - \text{CLN}) = 1.0/P$	C19693
C		C19694
C	IS MADE IN THE NUMERATOR WHERE CON IS AN ARBITRARY CONSTANT	C19695
C	NOT EQUAL TO A ROOT OF THE NUMERATOR. IT HAS BEEN ARBITRARILY	C19696
C	SET AS	C19697
C	$\text{CON} = -\text{USQRT}(3.0)$	C19698
C	IN DISCUS	C19699
C		C19700
C	THIS EVENTUALLY LEADS TO THE NEED TO DETERMINE THE ROOTS OF A	C19701
C	DETERMINANT EXPRESSION WHICH IS OF THE FORM	C19702
C		C19703
C	$\text{DET}(I*P - D)$	C19704
C		C19705
C	WHERE D IS A DERIVABLE NZ BY NZ MATRIX, SEE EQ III-41 VOL 1.	C19706
C	ALL NULL ROOTS IMPLY ROOTS AT INFINITY OR A CHARACTERISTIC	C19707
C	POLYNOMIAL HAVING ORDER LESS THAN NZ. THUS:	C19708
C		C19709
C	NN = ORDER OF NUMERATOR	C19710
C	$= \text{NZ} - \text{NUMBER OF ZERO ROOTS OF } \text{DET}(I*P - D)$	C19711
C	AND	C19712
C	S(I) = I-TH ROOT OF NUMERATOR IS OBTAINED FROM	C19713
C		C19714
C	$S(I) = \text{CON} + 1.0/P(I)$ (COMPLEX ARITHMETIC)	C19715
C		C19716
C	WHERE	C19717
C	$P(I) = \text{I-TH NON-ZERO ROOT OF } \text{DET}(I*P - D)$	C19718
C		C19719
C	THE FOLLOWING PARAMETERS ARE PASSED TO NUMS FROM DYN540 AND	C19720
C	RETURNED. BY SETTING DEBUG(68) = .TRUE. SEVERAL MAY BE PRINTED	C19721
C	ALONG WITH THE DESCRIPTIVE CAPTIONS INDICATED.	C19722
C	A = INPUTTED NZ BY NZ REDUCED MATRIX 'AR' WHICH WAS SETUP IN	C19723
C	TFTYPE, ARRAY DESTROYED DURING COMPUTATION	C19724
C	C = DUMMY ARRAY USED AS WORK SPACE	C19725
C	B = COLUMN OF THE 'B' MATRIX SETUP IN Tftype	C19726
C	R1R = OUTPUT OF NUMS. THE NZRD REAL PARTS OF THE COMPLEX ROOTS	C19727
C	OF THE NUMERATOR	C19728
C	R1I = OUTPUT OF NUMS. THE NZRD IMAGINARY PARTS OF THE COMPLEX	C19729
C	ROOTS OF THE NUMERATOR	C19730
C	R2R = WORK AREA	C19731
C	R2I = WORK AREA	C19732
C	PTOL = TOLERANCE FACTOR ALL P-ROOTS LESS THAN PTOL SET TO ZERO	C19733
C	GAIN = OUTPUT, NUMERATOR GAIN SEE EQ III-48 VOL 1	C19734
C	IFLG = 0 IF ERROR ZERO NUMERATOR GAIN, =1 IF NO ERROR	C19735
C	NN = 1	C19736
C	NZRO = OUTPUT THE ORDER OF AND HENCE NUMBER OF NUMERATOR ROOTS	C19737
C	IY = COLUMN LOCATION IN 'AR' OF DESIRED OUTPUT SIGNAL FOR	C19738
C	TRANSFER FUNCTION	C19739
C	NA = DIMENSION OF 'AR' (DEFINED AS NZ ABOVE)	C19740
C		C19741
C	'ANUM (NA X NA)' = PRINTED IF THE ORDER OF THE NUMERATOR	C19742
C	EQUAL TO THAT OF THE DENOMINATOR. IT	C19743
C	IS THE (NZ BY NZ) AUGMENTED MATRIX	C19744
C	PASSED TO THE QR-ALGORITHM (NA=NZ)	C19745

```

C      * -D- (NA X NA) * = PRINTED IF THE ORDER OF THE NUMERATOR 019746
C      LESS THAN THAT OF THE DENOMINATOR. IT 019747
C      IS THE MATRIX 'D' DEFINED ABOVE USED 019748
C      TO OBTAIN THE ROOTS OF DET(I*P - D) 019749
C      * -RIR- (1 X NZRU) * = THE REAL PARTS OF THE NZRU COMPLEX 019750
C      ROOTS OF THE NUMERATOR 019751
C      * -RII- (1 X NZRU) * = THE IMAGINARY PARTS OF THE NZRU COMPLEX 019752
C      ROOTS OF THE NUMERATOR 019753
C      * DRATIC (1 X NA) * = THE NA PIVOT ELEMENTS OBTAINED IN THE 019754
C      PROCESS OF COMPUTING THE 'D' MATRIX 019755
C      DRVEC IN DET(I*P - D). THESE ARE STORED IN 019756
C      ARRAY DRVEC DIMENSIONED (IGASM) IN 019757
C      /DRATIO/ 019758
C      * GAIN (1 X 1) * = NUMERATOR ROOT GAIN 019759
C      019760
C      019761
C      PUT IN STANDARD BODE FORM 019762
C      ----- 019763
C      019764
C      019765
C      FOR FREQUENCY RESPONSE CALCULATIONS IT IS CONVENIENT TO CONVERT 019766
C      POLES, ZEROS AND ROOT GAIN TO STANDARD BODE FORM. THAT IS TO 019767
C      WRITE THE TRANSFER FUNCTION IN TERMS OF TIME CONSTANTS, 019768
C      DAMPING ZETAS AND RESONANT FREQUENCIES. THE RESPONSE IS THEN 019769
C      EASILY GENERATED BY THE STANDARD SUBSTITUTION 019770
C      019771
C      S = J*OMEGA 019772
C      019773
C      AND EVALUATION OF THE TRANSFER FUNCTION EXPRESSION AT VARIOUS 019774
C      VALUES OF OMEGA 019775
C      019776
C      SUBROUTINE DCORRT IS USED TO CLASSIFY AND COUNT THE ROOTS OF THE 019777
C      DENOMINATOR AND NUMERATOR, WHILE SUBROUTINE DFORMB USES THE 019778
C      OUTPUT OF DCORRT TO COMPUTE TIME CONSTANT, DAMPING ZETA AND 019779
C      RESONANT FREQUENCY ARRAYS 019780
C      019781
C      THE NZ COMPLEX ROOTS OF THE DENOMINATOR ARE STORED TEMPORARILY IN 019782
C      THE ARRAYS RRD AND RID. THESE ARE PASSED TO DCORRT AND THE 019783
C      FOLLOWING COUNTS ARE RETURNED AND STORED IN /LCOUNT/: 019784
C      NOR = NUMBER OF REAL AND ZERO DENOMINATOR ROOTS 019785
C      NZR = NUMBER OF ZERO DENOMINATOR ROOTS 019786
C      ICP = NUMBER OF COMPLEX PAIRS IN DENOMINATOR ROOTS ARRAY 019787
C      THE DUMMY ARRAYS RLRT AND CMPR ARE ALSO LOADED IN DCORRT. 019788
C      THESE ARE IN TURN PASSED TO DFORMB WHERE DENOMINATOR TIME 019789
C      CONSTANTS, DAMPING ZETAS AND RESONANT FREQUENCIES ARE COMPUTED 019790
C      AND RETURNED IN THE FOLLOWING ARRAYS WHICH ARE STORED 019791
C      IN /LARRAY/: 019792
C      FBRD = TIME CONSTANT ARRAY FOR REAL AND ZERO DENOMINATOR ROOTS 019793
C      DIMENSIONED (LI)- ELEMENTS DEFINED ACCORDING TO 019794
C      FBRD(I) = 0.000 IF ELEMENT RLRT(I).EQ.0 019795
C      FBRD(I) = TIME CONSTANT ASSOCIATED WITH THE I-TH 019796
C      ELEMENT IN THE REAL DENOMINATOR ROOTS 019797
C      ARRAY RLRT (I=1,...,NOR) 019798
C      FDDC = DAMPING ZETA, RESONANT FREQUENCY ARRAY FOR DENOMINATOR 019799
C      ROOTS WHICH APPEAR AS COMPLEX PAIRS. DIMENSIONED (LI) 019800
C      FDDC(2*L-1) = DAMPING ZETA ASSOCIATED WITH THE L-TH 019801
C      COMPLEX ROOT PAIR IN THE DENOMINATOR 019802
C      ROOTS ARRAY CMPR 019803
C      FDDC(2*L) = RESONANT FREQUENCY ASSOCIATED WITH THE 019804
C      L-TH COMPLEX ROOT PAIR IN THE 019805

```

C		DENOMINATOR ROOTS ARRAY CMPR	019806
C		(L=1,2,....,ICD)	019807
C			019808
C	THE NN COMPLEX ROOTS OF THE NUMERATOR ARE STORED TEMPORARLY IN		019809
C	THE ARRAYS RRN AND RIN, THESE ARE PASSED TO UCQRRT AND THE		019810
C	FOLLOWING COUNTS ARE RETURNED AND STORED IN /LCOUNT/:		019811
C	NNR = NUMBER OF REAL AND ZERO NUMERATOR ROOTS		019812
C	NNZ = NUMBER OF ZERO NUMERATOR ROOTS		019813
C	ICN = NUMBER OF COMPLEX PAIRS IN NUMERATOR ROOTS ARRAY		019814
C	THE DUMMY ARRAYS RLRT AND CMPR ARE ALSO LOADED IN UCQRRT.		019815
C	THESE ARE IN TURN PASSED TO DFCRMB WHERE NUMERATOR TIME		019816
C	CONSTANTS, DAMPING ZETAS AND RESONANT FREQUENCIES ARE COMPUTED		019817
C	AND RETURNED IN THE FOLLOWING ARRAYS WHICH ARE STORED		019818
C	IN /LRARRAY/:		019819
C	FBNR = TIME CONSTANT ARRAY FOR REAL AND ZERO NUMERATOR ROOTS		019820
C	DIMENSIONED (L1), ELEMENTS DEFINED ACCORDING TO		019821
C	FBNR(1) = 0.000 IF ELEMENT RLRT(1).EQ.0		019822
C	FBNR(1) = TIME CONSTANT ASSOCIATED WITH THE I-TH		019823
C	ELEMENT IN THE REAL NUMERATOR ROOTS		019824
C	ARRAY RLRT (I=1,2,....,NNR)		019825
C	FBNC = DAMPING ZETA, RESONANT FREQUENCY ARRAY FOR NUMERATOR		019826
C	ROOTS WHICH APPEAR AS COMPLEX PAIRS, DIMENSIONED (L1)		019827
C	FBNC(2*L-1) = DAMPING ZETA ASSOCIATED WITH THE L-TH		019828
C	COMPLEX ROOT PAIR IN THE NUMERATOR		019829
C	ROOTS ARRAY CMPR		019830
C	FBNC(2*L) = RESONANT FREQUENCY ASSOCIATED WITH THE		019831
C	L-TH COMPLEX ROOT PAIR IN THE		019832
C	NUMERATOR ROOTS ARRAY CMPR		019833
C	(L=1,2,....,ICN)		019834
C			019835
C	BY SETTING DEBUG(80) = .TRUE. THE FOLLOWING CAPTIONS AND		019836
C	ASSOCIATED ARRAYS WILL BE PRINTED BEFORE EXIT FROM DFCRMB		019837
C	' OUTPUT MATRIX RLRT (1 X KR)' = REAL AND ZERO ROOTS ARRAY		019838
C	PASSED TO DFCRMB		019839
C	' OUTPUT MATRIX CMPR (1 X KK)' = COMPLEX ROOTS ARRAY PASSED		019840
C	TO DFCRMB		019841
C	' OUTPUT MATRIX FBR (1 X KR)' = TIME CONSTANT ARRAY RETURNED		019842
C	BY DFCRMB		019843
C	' OUTPUT MATRIX FBC (1 X KK)' = DAMPING ZETA AND RESONANT		019844
C	FREQUENCY ARRAY RETURNED BY		019845
C	DFCRMB		019846
C	' OUTPUT MATRIX ACCD (1 X 1)' = GAIN FOR NUMERATOR, 1.0 FOR		019847
C	DENOMINATOR RETURNED BY DFCRMB		019848
C	' OUTPUT MATRIX GB (1 X 2)' = PRODUCT OF ALL NON ZERO ROOTS		019849
C	RETURNED BY DFCRMB		019850
C			019851
C			019852
C	DATA SETUP FOR FREQUENCY RESPONSE AND ROOT LOCUS COMPUTATION		019853
C	-----		019854
C			019855
C	THE NUMERICAL ALGORITHMS USED TO COMPUTE THE TRANSFER FUNCTION'S		019856
C	FREQUENCY RESPONSE AND ROOT LOCUS DATA REQUIRE THAT THE POLE		019857
C	ZERO DATA OBTAINED BE PUT IN A PARTICULAR FORMAT. THIS IS		019858
C	ACCOMPLISHED BY SUBROUTINES TFF, IN WHICH THE ROOTS ARRAY 'R'		019859
C	IS SETUP, CANCEL, IN WHICH ROOTS COMMON TO THE NUMERATOR AND		019860
C	DENOMINATOR ARE CANCELLED AND RTOP, IN WHICH THE POLYNOMIAL		019861
C	COEFFICIENTS ARRAY 'RX' IS SETUP.		019862
C			019863
C			019864
C	TFF - SETUP ROOTS ARRAY 'R'		019865

```

C -----
C
C      = ROOTS ARRAY 'R' DIMENSIONED (2*L1+7) AND STORED
C      IN /LRROOT/. ALL NUMERATOR AND DENOMINATOR ROOT COUNTS,
C      REAL ROOTS, COMPLEX ROOTS AND DDC GAIN DATA ARE STORED
C      IN THE ROOTS ARRAY ACCORDING TO THE FOLLOWING FORMAT
C      R(1) = NUMBER OF REAL NON-ZERO NUMERATOR ROOTS
C      R(2) = NUMBER OF COMPLEX PAIRS IN NUMERATOR
C      R(3) = NUMBER OF ZERO ROOTS IN NUMERATOR
C      R(4) = NUMBER OF REAL NON-ZERO DENOMINATOR ROOTS
C      R(5) = NUMBER OF COMPLEX PAIRS IN DENOMINATOR
C      R(6) = NUMBER OF ZERO ROOTS IN DENOMINATOR
C      R(7) = DDC GAIN AS DEFINED BY EQUATIONS 111-48 AND 70
C            IN VOL 1. COMPUTED IN DYN540 FROM DATA GENERATED
C            IN DFCRMB AND RETURNED IN DUMMY ARRAY 'GB'
C
C      FOR I=3,9,...,L1 WHERE L1=7+R(1)
C          R(I) = THE R(1) TIME CONSTANTS ASSOCIATED WITH THE REAL
C                NON-ZERO NUMERATOR ROOTS
C
C      FOR I=L1+1,L1+3,...,L2-1 WHERE L2=7+R(1)+2*R(2)
C          R(I) = THE R(2) DAMPING ZETAS ASSOCIATED WITH THE
C                NUMERATOR ROOTS WHICH APPEAR AS COMPLEX PAIRS
C
C      FOR I=L1+2,L1+4,...,L2
C          R(I) = THE R(2) RESPECTIVE RESONANT FREQUENCIES
C                ASSOCIATED WITH THE NUMERATOR ROOTS WHICH APPEAR
C                AS COMPLEX PAIRS
C
C      FOR I=L2+1,...,L3 WHERE L3=7+R(1)+2*R(2)+R(4)
C          R(I) = THE R(4) TIME CONSTANTS ASSOCIATED WITH THE REAL
C                NON-ZERO DENOMINATOR ROOTS
C
C      FOR I=L3+1,L3+3,...,L4-1 WHERE L4=7+R(1)+2*R(2)+R(4)+2*R(5)
C          R(I) = THE R(5) DAMPING ZETAS ASSOCIATED WITH THE
C                DENOMINATOR ROOTS WHICH APPEAR AS COMPLEX PAIRS
C
C      FOR I=L3+2,L3+4,...,L4
C          R(I) = THE R(5) RESPECTIVE RESONANT FREQUENCIES
C                ASSOCIATED WITH THE DENOMINATOR ROOTS WHICH APPEAR
C                AS COMPLEX PAIRS
C
C      (TO PRINT SET DEBUG(63)=.TRUE., THE FOLLOWING CAPTION IS
C      PRINTED)
C
C      'SUBROUTINE TTFF ROOTS ARRAY IS' = THE FULL ROOTS ARRAY 'R' IS
C                                     PRINTED FOLLOWING THIS
C                                     CAPTION
C
C
C      CANCEL - CANCEL ROOTS COMMON TO NUMERATOR AND DENOMINATOR
C      -----
C
C      R      = ROOTS ARRAY 'R' DIMENSIONED (2*L1+7) AND STORED
C      IN /LRROOT/. THE ROOTS ARRAY 'R' AS SETUP IN TTFF IS
C      EXAMINED FOR ROOTS COMMON TO BOTH NUMERATOR AND
C      DENOMINATOR. A TOLERANCE FACTOR OF 10**(-7) IS USED TO
C      COMPARE THE TIME CONSTANTS AND A STRICT EQUALITY IS USED
C      TO COMPARE DAMPING ZETAS AND ASSOCIATED RESONANT
C      FREQUENCIES. WHEN COMMON ROOTS ARE FOUND THE APPROPRIATE
C      COUNTS ARE CHANGED AND THE ROOTS REMOVED FROM 'R'
C      (TO PRINT SET DEBUG(64)=.TRUE., THE FOLLOWING CAPTION IS
C      PRINTED)
C
C      'ROOT ARRAY OUT OF CANCEL IS' = THE FULL ROOTS ARRAY WITH ROOTS
C                                     COMMON TO NUMERATOR AND
C                                     DENOMINATOR CANCELLED IS PRINTED
C                                     FOLLOWING THIS CAPTION
C

```


C		019926
C	RTOP - CONSTRUCT POLYNOMIAL COEFFICIENTS ARRAY 'RX'	019927
C	-----	019928
C		019929
C	THE ALGORITHM USED FOR NUMERICAL COMPUTATION OF ROOT LOCUS DATA	019930
C	REQUIRES THAT THE TRANSFER FUNCTION BE GIVEN AS A RATIO OF	019931
C	POLYNOMIALS. THE COEFFICIENTS OF THE NUMERATOR AND DENOMINATOR	019932
C	POLYNOMIALS ARE COMPUTED IN RTOP DIRECTLY FROM THE DATA STORED	019933
C	IN THE ROOTS ARRAY 'R'	019934
C		019935
C	RX = POLYNOMIAL COEFFICIENTS ARRAY 'RX', DIMENSIONED(4*L1+14)	019936
C	AND STORED IN /LRCT/. THE DEGREE AND THE COEFFICIENTS	019937
C	OF BOTH THE NUMERATOR AND DENOMINATOR POLYNOMIALS ARE	019938
C	STORED AS FOLLOWS (RX IS RETURNED TO DYN40 IN DUMMY	019939
C	ARRAY POLY)	019940
C	RX(1) = THE DEGREE OF THE NUMERATOR POLYNOMIAL	019941
C	RX(2) = THE DEGREE OF THE DENOMINATOR POLYNOMIAL	019942
C	FOR I=3,4,...,3+RX(1)	019943
C	RX(I) = COEFFICIENT OF THE S**(I-3) TERM IN THE	019944
C	NUMERATOR POLYNOMIAL	019945
C	FOR I=4+RX(1),5+RX(1),...,4+RX(2)+RX(1)	019946
C	RX(I) = COEFFICIENT OF THE S**(I-(4+RX(1))) TERM IN THE	019947
C	DENOMINATOR POLYNOMIAL	019948
C	THE CONTENTS OF THE POLYNOMIAL COEFFICIENTS ARRAY ARE PRINTED	019949
C	UNDER THE FOLLOWING CAPTIONS ALWAYS	019950
C	* OUTPUT MATRIX PDEN (1 X ND) * = THE ND=RX(1)+1 COEFFICIENTS	019951
C	ASSOCIATED WITH THE DENOMINATOR POLYNOMIAL	019952
C	FOR ASCENDING POWERS OF S	019953
C	* OUTPUT MATRIX PNUM (1 X NP) * = THE NP=RX(2)+1 COEFFICIENTS	019954
C	ASSOCIATED WITH THE NUMERATOR POLYNOMIAL	019955
C	FOR ASCENDING POWERS OF S	019956
C		019957
C		019958
C	FREQUENCY RESPONSE, EIGENVECTOR AND ROOT LOCUS COMPUTATION	019959
C	-----	019960
C		019961
C	ALL DATA REQUIRED TO CONSTRUCT THE USER REQUESTED TRANSFER	019962
C	FUNCTION IS IN THE DATA ARRAYS R AND RX STORED IN /LRCT/.	019963
C	THE TRANSFER FUNCTION MAY BE EXPRESSED IN STANDARD BODE	019964
C	FACTORED POLYNOMIAL FORM FROM THE DATA IN 'R' OR IT MAY BE	019965
C	EXPRESSED IN AN UNFACTORED POLYNOMIAL FORM WITH THE DATA	019966
C	IN 'RX'.	019967
C		019968
C	FOUR GRAPHICAL METHODS OF TRANSFER FUNCTION ANALYSIS ARE AVAILABLE	019969
C	FOR USER REQUEST IN DISCOS. THE DATA ASSOCIATED WITH EACH	019970
C	MAY BE EITHER PLOTTED DIRECTLY VIA USE OF THE SC4020 PLOT	019971
C	PACKAGE OR DUMPED ON TAPE FOR PROCESSING OUTSIDE OF DISCOS.	019972
C	THE GRAPHICAL REPRESENTATIONS WHICH ARE AVAILABLE FOR USER	019973
C	REQUEST ARE:	019974
C	1) BODE-PLOT REPRESENTATIONS	019975
C	2) NICHOLS CHARTS	019976
C	3) NYQUIST DIAGRAMS	019977
C	4) ROOT-LOCUS PLOTS	019978
C		019979
C	EIGENVECTOR ANALYSIS IN CONJUNCTION WITH FREQUENCY RESPONSE	019980
C	ANALYSIS IS NOT A COMMON PRACTISE; HOWEVER, FOR COMPLEX HIGHLY	019981
C	COUPLED MULTI-RIGID AND FLEXIBLE BODY SYSTEMS IT BECOMES A	019982
C	NECESSITY. SYSTEM ROOTS OFTEN SHIFT SIGNIFICANTLY FROM BODY	019983
C	AND CONTROL FREQUENCIES AND HENCE ARE IMPOSSIBLE TO PHYSICALLY	019984
C	INTERPRET. EIGENVECTOR ANALYSIS OF THE CHARACTERISTIC	019985

C	MATRIX AR ASSOCIATED WITH THE TRANSFER FUNCTION UNDER STUDY	019986
C	CAN BE USED TO DETERMINE HOW AND TO WHAT DEGREE VARIOUS BODY	019987
C	AND CONTROL SYSTEM MODES AND FREQUENCIES BLEND TOGETHER TO	019988
C	FORM THE TOTAL SYSTEM ROOTS AND ASSOCIATED MODES OF VIBRATION	019989
C	(EIGENVECTORS)	019990
C		019991
C	TO ACTIVATE THE EIGENVECTOR ANALYSIS CAPABILITY THE USER MUST	019992
C	INPUT THE CODE WORD:	019993
C		019994
C	LEIGV = 4HEIGV ON THE SAME DATA CARD WITH LPNAME AND LPTAPE	019995
C		019996
C	UPON RECOGNITION THE COMPUTER READS IN FQMIN AND FQMAX WHICH	019997
C	DEFINE WHICH COMPLEX ROOTS OF THE CHARACTERISTIC MATRIX AR	019998
C	REQUIRE ASSOCIATED EIGENVECTOR COMPUTATION. ALL COMPLEX ROOTS	019999
C	HAVING IMAGINARY PART .GE. FQMIN .AND. .LE. FQMAX ARE FED INTO	020000
C	SUBROUTINE EIGVEC WHICH COMPUTES THE ASSOCIATED EIGENVECTORS	020001
C		020002
C	*** PROVIDING THAT ROOTS ARE NOT MULTIPLE ROOTS ***	020003
C		020004
C	SOMEDAY I'LL FIND A BETTER ROUTINE FOR MULTIPLE ROOTS H.P.F	020005
C		020006
C		020007
C	THE OUTPUT MATRICES PRINTED AFTER THE FOLLOWING CAPTIONS ARE:	020008
C		020009
C	'OUTPUT MATRIX ROUTRE IS' = REAL PARTS OF THE ROOTS OF THE	020010
C	CHARACTERISTIC MATRIX AR FOR WHICH	020011
C	EIGENVECTORS WILL BE COMPUTED	020012
C	'OUTPUT MATRIX ROUTIE IS' = IMAGINARY PARTS OF THE ROOTS OF THE	020013
C	CHARACTERISTIC MATRIX AR FOR WHICH	020014
C	EIGENVECTORS WILL BE COMPUTED	020015
C	'EIGENVALUE =' = THE EIGENVALUE ASSOCIATED WITH THE	020016
C	FOLLOWING EIGENVECTOR	020017
C	'EIGENVECTOR OF NORMAL MARKIX' = THE NZ ELEMENTS OF THE	020018
C	EIGENVECTOR ASSOCIATED WITH THE	020019
C	ABOVE EIGENVALUE. THE DEGREES OF	020020
C	FREEDOM CONTAINED IN THE	020021
C	EIGENVECTOR ARE THOSE ASSOCIATED	020022
C	WITH THE Z' ARRAY CONSTRUCTED	020023
C	ABOVE FOR THE OPTION 'ITYPE' USED	020024
C	IN SUBROUTINE TTYPE	020025
C	'MORE THAN 10 LOOPS FOR = PROBABLY ROOT SENT TO EIGVEC IS	020026
C	EIGENVECTOR OF ... DIFFERENCE A MULTIPLE ROOT. EIGENVECTORS	020027
C	OF ...' CANNOT BE FOUND. EIGENVECTOR LIST	020028
C	FOLLOWING PROBABLY GARBAGE	020029
C	NOTE *** TO FIND THE DEGREE OF FREEDOM ASSOCIATED WITH EACH	020030
C	ELEMENT OF THE EIGENVECTOR USE 'THE TRANSFORMED	020031
C	STATE VECTOR CORRELATION ARRAY' DEFINED ABOVE	020032
C	AND THE LOGIC LOOPS USED FOR THE PARTICULAR ITYPE	020033
C	TRANSFER FUNCTION EVALUATED BY TTYPE	020034
C		020035
C	THE USER MUST INPUT VARIOUS CODE WORDS TO SPECIFY WHICH TYPE OF	020036
C	DISPLAY IS REQUESTED AND IF THE GRAPHIC DISPLAY DATA IS TO BE	020037
C	COMP. THE FOLLOWING CONTROL WORDS ARE READ IN DYN540	020038
C	LPNAME = DISPLAY TYPE CODE WORD INTEGER ARRAY DIMENSIONED (5).	020039
C	AFTER EACH REQUESTED TRANSFER FUNCTION IS CONSTRUCTED	020040
C	THE FULL 5 ELEMENT ARRAY LPNAME IS READ. THE FOLLOWING	020041
C	CODES ARE RECOGNIZED TO MEAN:	020042
C	LPNAME(1) = 4H = NO DISPLAY REQUESTED	020043
C	1=1,....5 4HRODE = GET BODY DISPLAY ONLY	020044
C	WHICH = GET NICHOLS DISPLAY ONLY	020045

```

C          4HNYQU = GET NYQUIST DISPLAY ONLY                                020046
C          4HNINY = GET NICHOLS AND NYQUIST DISPLAYS                      020047
C          4HBUNN = GET BODE, NICHOLS AND NYQUIST DISPLAYS                020048
C          4HROOT = GET ROOT LOCUS DISPLAY                               020049
C  LPTAPE = DISPLAY DATA DUMP CODE. THIS CODE IS READ FROM THE SAME      020050
C          INPUT DATA CARD USED TO DEFINE THE LPNAME ARRAY. ALL RAW      020051
C          DISPLAY DATA IS COMPUTED IN EITHER SFREQ2 OR RLUCUS AND        020052
C          STORED IN /PSTUFF/. IMMEDIATELY UPON RETURN TO DYN540          020053
C          FROM SFREQ2 OR RLUCUS THE FOLLOWING CODED CHECK IS MADE:       020054
C          IF (LPTAPE.NE.4H7TRK) GO TO 1                                  020055
C          WRITE(12,100) KSAVE                                           020056
C          WRITE(12,101) (SAVE(I),SAVEP(I),                               020057
C          *          SAVE(I),SAVEA(I),I=1,KSAVE)                        020058
C          1 CONTINUE                                                    020059
C          100 FORMAT (15)                                                020060
C          101 FORMAT (4D18.8)                                           020061
C                                                                           020062
C          FREQUENCY RESPONSE PROCESSING                                020063
C          -----                                                        020064
C                                                                           020065
C  THE FREQUENCY RESPONSE PROCESSING SECTION OF DYN540 IS ENTERED        020066
C          IF (LPNAME(1).EQ.4HBODE.OR.                                    020067
C          LPNAME(1).EQ.4HNICH.OR.                                        020068
C          LPNAME(1).EQ.4HNYQU.OR.                                       020069
C          LPNAME(1).EQ.4HNINY.OR.                                       020070
C          LPNAME(1).EQ.4HBUNN)                                          020071
C          WHERE I=1,2,...,5. SUBROUTINE SFREQ2 DOES ALL OF THE BASIC     020072
C          COMPUTATION REQUIRED. THE USER HAS CONTROL OF THE FREQUENCY    020073
C          RANGE OVER WHICH FREQUENCY RESPONSE DATA IS TO BE COMPUTED.   020074
C          THIS IS DONE BY USER SPECIFICATION OF TWO RANGE PARAMETERS:    020075
C          FMIN = LOWER LIMIT FOR FREQUENCY RESPONSE COMPUTATION IN SFREQ2 020076
C          A REAL*4 CONSTANT STORED IN /LBDATA/ DEFINED FOR EACH          020077
C          LPNAME(I),I=1,5 SATISFYING THE ABOVE CRITERION                020078
C          FMAX = UPPER LIMIT FOR FREQUENCY RESPONSE COMPUTATION IN SFREQ2 020079
C          A REAL*4 CONSTANT STORED IN /LBDATA/ DEFINED FOR EACH          020080
C          LPNAME(I),I=1,5 SATISFYING THE ABOVE CRITERION                020081
C                                                                           020082
C          IF EITHER BODE PLOTS OR NICHOLS CHARTS ARE TO BE GENERATED VIA 020083
C          THE SC4020 PLOT PACKAGE ROUTINES A RANGE FOR DB MAGNITUDE OF   020084
C          THE FREQUENCY RESPONSE MUST BE GIVEN. THIS IS DONE BY USER    020085
C          SPECIFICATION OF TWO RANGE PARAMETERS:                          020086
C          DBMIN = LOWER LIMIT FOR DB MAGNITUDE OF FREQUENCY RESPONSE DATA 020087
C          TO BE PLOTTED. A REAL*4 CONSTANT STORED IN /LBDATA/. USER      020088
C          DEFINED FOR EACH LPNAME(I),I=1,5 SATISFYING THE ABOVE          020089
C          CRITERION                                                       020090
C          DBMAX = UPPER LIMIT FOR DB MAGNITUDE OF FREQUENCY RESPONSE DATA 020091
C          TO BE PLOTTED. A REAL*4 CONSTANT STORED IN /LBDATA/. USER      020092
C          DEFINED FOR EACH LPNAME(I),I=1,5 SATISFYING THE ABOVE          020093
C          CRITERION                                                       020094
C                                                                           020095
C          IF NYQUIST DIAGRAMS ARE TO BE GENERATED VIA THE SC4020 PLOT    020096
C          PACKAGE ROUTINE AN AMPLITUDE RANGE FOR THE FREQUENCY RESPONSE   020097
C          MUST BE GIVEN. THIS IS DONE BY USER SPECIFICATION OF TWO       020098
C          RANGE PARAMETERS                                                020099
C          AMIN = LOWER LIMIT FOR AMPLITUDE OF FREQUENCY RESPONSE DATA TO 020100
C          BE PLOTTED. A REAL*4 CONSTANT STORED LOCALLY USER              020101
C          DEFINED FOR EACH LPNAME(I),I=1,5 SATISFYING THE ABOVE          020102
C          CRITERION                                                       020103
C          AMAX = UPPER LIMIT FOR AMPLITUDE OF FREQUENCY RESPONSE DATA TO 020104
C          020105

```

C	BE PLOTTED. A REAL*4 CONSTANT STORED LOCALLY. USER	020106
C	DEFINES FOR EACH LPNAME(I), I=1,5 SATISFYING THE ABOVE	020107
C	CRITERION	020108
C		020109
C		020110
C	FREQUENCY RESPONSE DATA	020111
C	-----	020112
C		020113
C	BODE, NYQUIST AND NICHOLS FREQUENCY RESPONSE DISPLAYS SIMPLY	020114
C	PRESENT THE SAME DATA IN DIFFERENT WAYS. ALL FREQUENCY	020115
C	RESPONSE DATA IS GENERATED IN SUBROUTINE SFREQ2 AND STORED	020116
C	IN /PSTUFF/. VARIABLE INCREMENTING TECHNIQUES ARE USED IN	020117
C	THE FREQUENCY BAND FMIN TO FMAX TO ACHIEVE ADEQUATE RESOLUTION	020118
C	AROUND SYSTEM NATURAL FREQUENCIES. THE FOLLOWING REAL*4 ARRAYS	020119
C	DIMENSIONED (LS) AND STORED IN /PSTUFF/ CONTAIN THE FOLLOWING	020120
C	DATA WHEN LOADED VIA SFREQ2	020121
C	KSAVE = INTEGER CONSTANT, DEFINES TOTAL NUMBER OF FREQUENCY	020122
C	RESPONSE DATA POINTS IN EACH OF THE /PSTUFF/ ARRAYS	020123
C		020124
C	VARIABLE INCREMENTING TECHNIQUES ARE USED TO CHOOSE THE VALUES	020125
C	OF 'OMEGA' AT WHICH THE TRANSFER FUNCTION 'TF' (IN BODE FORM)	020126
C	IS TO BE EVALUATED. LET:	020127
C	I = SQRT(-1.0)	020128
C	COMPLEX VALUE OF TRANSFER FUNCTION 'TF' AT FREQUENCY 'OMEGA'	020129
C	TF(I*OMEGA) = ALPHA + I*BETA	020130
C	AMPLITUDE	020131
C	AR = DSQRT(ALPHA**2 + BETA**2)	020132
C	PHASE (DEGREES)	020133
C	DPHI = DATAN2(BETA,ALPHA)*57.2958	020134
C	0 < DPHI < 360.0	020135
C	AMPLITUDE IN DB.	020136
C	DBELL = 8.68588961*DLDB(AH)	020137
C	8.68588961 = 20*LOG(E)	020138
C	THEN:	020139
C	SAVEC = ALL VALUES OF 'OMEGA' FOR WHICH TRANSFER FUNCTION IS	020140
C	EVALUATED	020141
C	SAVEA = ALL RESPECTIVE VALUES OF AMPLITUDE 'AR'	020142
C	SAVLP = ALL RESPECTIVE VALUES OF PHASE IN DEGREES 'DPHI'	020143
C	SAVEB = ALL RESPECTIVE VALUES OF AMPLITUDE IN DB 'DBELL'.	020144
C		020145
C		020146
C	ROOT LOCUS PROCESSING	020147
C	-----	020148
C		020149
C	THE ROOT LOCUS PROCESSING SECTION OF DISCUS IN DYN540 IS ENTERED	020150
C	IF(LPNAME(I).EQ.4HROOT)	020151
C	WHERE I=1,2,....,5. SUBROUTINE RLOCUS DOES ALL OF THE BASIC	020152
C	COMPUTATION REQUIRED FOR ROOT LOCUS PLOT GENERATION.	020153
C	TO EFFECTIVELY APPLY THIS CAPABILITY IN DISCUS THE USER MUST	020154
C	BE COGNIZANT OF THE BASIC TECHNIQUE USED FOR ROOT LOCUS	020155
C	GENERATION:	020156
C	1) RLOCUS ACCEPTS A NUMERATOR POLYNOMIAL N(S) AND A	020157
C	DENOMINATOR POLYNOMIAL D(S). IT ASSUMES THAT THE RATIO	020158
C	OF THESE NUMERICALLY DERIVED POLYNOMIALS IS EITHER AN	020159
C	OPEN OR QUASI-OPEN LOOP TRANSFER FUNCTION. DISCUS DOES NOT	020160
C	CROSS CHECK THIS. RLOCUS BLINDLY ACCEPTS ANY PAIR OF	020161
C	POLYNOMIALS N(S) AND D(S).	020162
C	2) LET:	020163
C	N(S)	020164
C	G(S)*H(S) = K * ---- = OPEN LOOP TRANSFER FUNCTION	020165

```

C                                     C(S)                                020166
C      THEN                                                                    020167
C                                     C(S)                                020168
C                                     G(S)                                020169
C      ----- = ----- = CLOSED LOOP TRANSFER FUNCTION 020170
C      R(S)  1 + G(S)*H(S) 020171
C      AND                                                                    020172
C                                     N(S)                                020173
C      1 + K*----- = 0 = CHARACTERISTIC EQUATION. THE LOCUS 020174
C                                     D(S)                                020175
C                                     OF THE ROOTS PLOTTED IN THE COMPLEX 020176
C                                     S-PLANE AS A FUNCTION OF THE OPEN 020177
C                                     LOOP GAIN FACTOR K IS THE DESIRED 020178
C                                     ROOT LOCUS 020179
C      3) FOR COMPLEX HIGHER ORDER SYSTEMS THE COMPUTATION OF ALL 020180
C      ROOT LOCI IS UNNECESSARY. CONSEQUENTLY DISCUS USES AN 020181
C      ITERATIVE PROCEDURE WHICH REQUIRES THAT THE USER DEFINE 020182
C      THE PARTICULAR OPEN LOOP TRANSFER FUNCTION POLE OR ZERO 020183
C      WHICH WILL BE THE STARTING POINT FOR THE ALGORITHM. ONCE 020184
C      CHOSEN THE LOCUS FROM OPEN LOOP POLE (K=0) TO OPEN LOOP 020185
C      ZERO (K=INFINITY) IS EVALUATED. THE FOLLOWING PARAMETERS 020186
C      ARE SET BY INPUT DATA CARDS: 020187
C      NRLOC = TOTAL NUMBER OF ROOT LOCI TO BE COMPUTED FOR PARTICULAR 020188
C      OPEN LOOP TRANSFER FUNCTION DEFINED BY N(S) AND D(S) 020189
C      IJM = LOCI STARTING POINT ARRAY. A (2 BY NRLOC) INTEGER ARRAY 020190
C      STORED LOCALLY. FOR J=1,2,...,NRLOC 020191
C      ISNIM = IJM(1,J) = 1 IF STARTING POINT OF THE J-TH LOCUS TO BE 020192
C      COMPUTED IS AN OPEN LOOP ZERO; THAT IS, A 020193
C      ROOT OF N(S) 020194
C      = 2 IF STARTING POINT OF THE J-TH LOCUS TO BE 020195
C      COMPUTED IS AN OPEN LOOP POLE; THAT IS, A 020196
C      ROOT OF D(S) 020197
C      ALL ROOTS OF THE NUMERATOR N(S) ARE LISTED UNDER THE 020198
C      CAPTION 020199
C      *NUM* 020200
C      *NO REAL PART IMAGINARY PART* 020201
C      ALL ROOTS OF THE DENOMINATOR D(S) ARE LISTED UNDER THE 020202
C      CAPTION 020203
C      *DEM* 020204
C      *NO REAL PART IMAGINARY PART* 020205
C      JJ = IJM(2,J) = THE INTEGER NUMBER, TAKEN FROM THE DATA 020206
C      PRINTED UNDER THE ABOVE CAPTIONS, ASSOCIATED 020207
C      WITH THE PARTICULAR POLE OR ZERO WHICH IS TO 020208
C      BE USED AS THE STARTING POINT FOR THE J-TH 020209
C      ROOT LOCUS GENERATED 020210
C      4) THE ROOT LOCUS ITERATION ALGORITHM IS DISCUSSED IN VOL 1 020211
C      APPENDIX L. SIX ADDITIONAL PARAMETERS MUST BE USER 020212
C      SUPPLIED FOR EACH ROOT LOCUS. THE FOLLOWING INPUT ARRAY 020213
C      SETS THE ITERATION CONTROL PARAMETERS 020214
C      #1 = ITERATION CONTROL DATA ARRAY. A (5 BY NRLOC) REAL*8 ARRAY 020215
C      STORED IN THE WORK AREA IN /LWORK1/. FOR J=1,2,...,NRLOC 020216
C      THETA0 = W1(1,J) = INITIAL SEARCH ANGLE FOR J-TH LOCUS. 020217
C      NOMINALLY EQUAL TO 180.000 (DEGREES) 020218
C      SCL = W1(2,J) = SCALE FACTOR FOR J-TH LOCUS. NOMINALLY EQUAL 020219
C      TO 1.000. IF (W1(2,J) .LT. 0.0) SCALE FACTOR 020220
C      WILL BE COMPUTED 020221
C      ALGC = W1(3,J) = PHASE CONTROL PARAMETER EQUALS 1.000 IF GAIN 020222
C      FACTOR POSITIVE; EQUALS -1.0 IF GAIN FACTOR K 020223
C      NEGATIVE 020224
C      XMIN = W1(4,J) = MINIMUM ADMISSIBLE REAL ROOT VALUE 020225
C      XMAX = W1(5,J) = MAXIMUM ADMISSIBLE REAL ROOT VALUE

```

```

C      YMAX = W(6,J) = MAXIMUM ADMISSIBLE IMAGINARY ROOT VALUE      020226
C      5) BRIEFLY, THE ALGORITHM IS STARTED AT A KNOWN POINT ON THE 020227
C      LOCUS, NAMELY AN OPEN LOOP POLE OR ZERO. A SMALL CIRCLE      020228
C      IN THE S-PLANE ABOUT THE STARTING POINT (DIAMETER VARIED     020229
C      INTERNALLY) IS DEFINED AND A SEARCH MADE TO FIND WHERE       020230
C      THE LOCUS INTERSECTS THE CIRCLE. THE INTERSECTION POINT     020231
C      IS A POINT ON THE LOCUS, A CIRCLE IS DEFINED ABOUT THE NEW   020232
C      POINT AND THE PROCEDURE REPEATED. SINCE                      020233
C                                                                    020234
C      
$$K * \frac{N(S)}{D(S)} = -1$$
      020235
C      020236
C      CIRCLE INTERSECTION IS DEFINE BY THAT VALUE OF S FOR WHICH  020237
C      THE PHASE ANGLE OF (N(S)/D(S)) = 180 DEGREES. THE OPEN      020238
C      LOOP GAIN FACTOR K CAN THEN BE COMPUTED DIRECTLY FROM THE    020239
C      EQUATION  $K = \frac{DABS(D(S))}{DABS(N(S))}$       020240
C      020241
C      020242
C      020243
C      020244
C      020245
C      020246
C      020247
C      020248
C      020249
C      020250
C      020251
C      020252
C      020253
C      020254
C      020255
C      020256
C      020257
C      020258
C      020259
C      020260
C      020261
C      020262
C      020263
C      020264
C      020265
C      020266
C      020267
C      020268
C      020269
C      020270
C      020271
C      020272
C      020273
C      020274
C      020275
C      020276
C      020277
C      020278
C      020279
C      020280
C      020281
C      020282
C      020283
C      020284
C      020285

```

ALL ROOT LOCUS COMPUTATION IS DONE IN SUBROUTINE RLOCUS. THE USER
 DEFINES THE NUMBER OF ROOT LOCI OF INTEREST FOR EACH TRANSFER
 FUNCTION AND A STARTING POINT FOR THE ALGORITHM ON EACH
 RESPECTIVE LOCUS. DYN540 CALLS RLOCUS EXACTLY NRLOC TIMES. EACH
 CALL TO RLOCUS REQUESTS THE COMPUTATION OF ANOTHER ROOT LOCUS
 ALL DATA IS STORED IN /PSTUFF/ AND MAY BE OUTPUTTED ON TAPE
 IF (LPTAPE.EQ.40/TRK). THE FOLLOWING ARRAYS IN /PSTUFF/ ARE
 USED BY RLOCUS
 KSAVE = TOTAL NUMBER OF POINTS ON THE PARTICULAR ROOT LOCUS
 GENERATED
 SAVEC = ALL REAL COORDINATES OF THE KSAVE POINTS ON THE ROOT
 LOCUS GENERATED
 SAVEF = ALL IMAGINARY COORDINATES OF THE RESPECTIVE KSAVE POINTS
 ON THE ROOT LOCUS GENERATED
 IN DYN540 AFTER EACH RETURN FROM RLOCUS THE DATA IN /PSTUFF/
 IS EITHER PLOTTED OR DUMPED ON TAPE UNIT 12 OR BOTH
 EVERY TIME RLOCUS IS ENTERED SEVERAL DATA SETS ARE PRINTED FOR THE
 USER. THE DATA PRINTED ALONG WITH THE FOLLOWING CAPTIONS ARE:
 *N(S) = = POLYNOMIAL FORM OF THE
 + (A0) + (A1)*S**1 NUMERATOR OF THE TRANSFER
 + (A2)*S**2 + (A3)*S**3 + ... FUNCTION FOR WHICH THE ROOT
 LOCUS IS TO BE COMPUTED. A0,A1,
 A2,... ARE THE COEFFICIENTS
 *D(S) = = POLYNOMIAL FORM OF THE
 + (B0) + (B1)*S**1 DENOMINATOR OF THE TRANSFER
 + (B2)*S**2 + (B3)*S**3 + ... FUNCTION FOR WHICH THE ROOT
 LOCUS IS TO BE COMPUTED. B0,B1,
 B2,... ARE THE COEFFICIENTS
 *STARTING POINT = = THE REAL AND IMAGINARY PARTS OF
 (SR) + I*(SI) THE STARTING POINT ON THE ROOT
 LOCUS TO BE GENERATED
 *SCAN LIMITS: = SCAN LIMITS FOR THE REAL PART X
 XMIN < X < XMAX AND IMAGINARY PART Y FOR EACH
 -YMAX < Y < YMAX POINT ON THE ROOT LOCUS
 COMPUTATION TERMINATED IF A
 LIMIT EXCEEDED
 * 180 DEGREE LOCUS = OPEN LOOP GAIN K POSITIVE

C	' 0 DEGREE LOCUS'	= OPEN LOOP GAIN K NEGATIVE	020286
C	'SCALE FACTOR = '	= SCALE FACTOR USED DURING	020287
C		COMPUTATION OF ROOT LOCUS	020288
C	'GAIN ROOTS ERROR'	= LISTED UNDER THESE HEADINGS ARE	020289
C		FOR EACH POINT FOUND ON THE	020290
C		ROOT LOCUS: THE OPEN LOOP GAIN	020291
C		FACTOR K, THE REAL AND IMAGINARY	020292
C		PART OF THE ROOT, AND THE REAL	020293
C		AND IMAGINARY PARTS OF THE ERROR	020294
C		MEASURE COMPUTED INTERNALLY FOR	020295
C		THE ROOT	020296
C			020297
C			020298
C	LINEARIZED TIME DOMAIN ANALYSIS		020299
C	-----		020300
C			020301
C	THE LINEARIZED TIME DOMAIN ANALYSIS PORTION OF THE PROGRAM IS		020302
C	ENTERED ONLY		020303
C	IF(LNAM.EQ.4*TIME)		020304
C	WHERE		020305
C	LNAM = INPUT CONTROL WORD DEFINED ABOVE UNDER 'FREQUENCY		020306
C	DOMAIN ANALYSIS'		020307
C	THE LINEARIZED DIFFERENTIAL EQUATIONS OF MOTION WHICH ARE		020308
C	NUMERICALLY SOLVED IN SUBROUTINE LTRESF ARE OF THE FORM:		020309
C			020310
C	ZDT(I,T) = K**(-1)*H*R(I,J)*Z(J,T) + B(I,J)*U(J,T)		020311
C			020312
C	WHERE THE ELEMENTS CONTAINED IN MATRICES		020313
C			020314
C	ZDT(I,T)		020315
C	Z(I,T)		020316
C	K**(-1)*H*R(I,J)		020317
C			020318
C	ARE DEFINED UNDER THE TITLE 'SIMILARITY TRANSFORMATION' ABOVE,		020319
C	AND THE QUANTITY		020320
C	B(I,J)*U(J,T) = USER SUPPLIED EXTERNAL SYSTEM INPUT ACTING		020321
C	ON THE I-TH PERTURBATION VARIABLE. THE USER		020322
C	MUST CODE THIS QUANTITY IN SUBROUTINE LTRCRL		020323
C			020324
C	ON PAGE 85, VOL 1 THE RECURSIVE FORMULAR USED FOR		020325
C	NUMERICAL INTEGRATION OF THE LINEARIZED EQUATIONS IS DEFINED.		020326
C	INITIAL CONDITIONS FOR ALL PERTURBATION VARIABLES ARE ASSUMED		020327
C	TO BE ZERO. NOTE THAT THESE EQUATIONS INCLUDE ALL NAUX SENSOR		020328
C	AND TORQUE SIGNAL EQUATIONS AS PERTURBATION VARIABLES.		020329
C	SUBROUTINE LPRINT IS ENTERED EVERY NOPRINT INTEGRATION CYCLES.		020330
C	A COMPLETE SELF EXPLANATORY PRINTOUT OF ALL PERTURBATION		020331
C	VARIABLES IS PROVIDED		020332
C			020333
C			020334
C	TAPE NUMBERS AND NUMERIC CONSTANTS		020335
C	-----		020336
C	THE FOLLOWING PARAMETERS DEFINE THE TAPE NUMBERS WHICH IN THE JCL		020337
C	ARE ALLOCATED FOR USE AS SCRATCH TAPES (SHOULD BE DISK SPACE)		020338
C	THEY ARE INTEGER*4 CONSTANTS, SET IN MAIN, STORED IN /TAPEND/		020339
C	NTAPE1 = 1		020340
C	NTAPE2 = 2		020341
C	NTAPE3 = 3		020342
C			020343
C	THE FOLLOWING ARE THE OTHER UNITS USED IN DISCUS		020344
C	NIT = 5 (CARD READER) DEFINED BY DATA STATEMENT		020345

C	PLT = 6 (LINE PRINTER) DEFINED BY DATA STATEMENT	020346
C		020347
C	THE FOLLOWING UNITS ARE ALLOCATED IN THE USFC 300-91 VERSION	020348
C	OF DISCUS, THEY BE USED AT THE USER'S DISCRETION	020349
C	UNIT 10 = 9-TRACK TAPE FOR USER TAPE INPUT	020350
C	UNIT 11 = 9-TRACK TAPE FOR USER TAPE OUTPUT	020351
C	UNIT 12 = 7-TRACK TAPE FOR USER TAPE OUTPUT	020352
C	UNIT 14 = 9-TRACK TAPE FOR USER FLEXIBLE BODY DATA INPUT	020353
C		020354
C	THE FOLLOWING PARAMETERS ARE REAL*8 NUMERIC CONSTANTS SET IN MAIN	020355
C	AND STORED IN /NUMBERS/	020356
C	ZRC = 0.000	020357
C	CNE = 1.000	020358
C	TWC = 1.000	020359
C	TRCS = 3.000	020360
C		020361
C		020362
C	PRINTOUT EMBELLISHMENT PARAMETERS	020363
C	-----	020364
C	THE FOLLOWING PARAMETERS ARE INPUTTED IN SUBROUTINE START	020365
C	AND STORED IN /ESTRT1/ AS REAL*8 ARRAYS	020366
C	UNAME = USER NAME READ WITH 3A6 FORMAT, DIMENSIONED (3)	020367
C	TITLE1 = FIRST TITLE TO APPEAR ON EACH DATA OUTPUT PAGE OF	020368
C	PRINTOUT, READ WITH 12A6 FORMAT, DIMENSIONED (12).	020369
C	TITLE2 = SECOND TITLE TO APPEAR ON EACH DATA OUTPUT PAGE OF	020370
C	PRINTOUT, READ WITH 12A6 FORMAT, DIMENSIONED (12)	020371
C		020372
C	THE FOLLOWING PARAMETERS ARE EITHER READ OR INITIALIZED IN	020373
C	SUBROUTINE START AND STORED IN /LSTART/	020374
C	IRUNNC = RUN NUMBER, READ IN, IN START AS A REAL*8 PARAMETER WITH	020375
C	FORMAT A6. IF IRUNNC.EQ.0/DEBUG THE DEBUG PRINT OPTIONS	020376
C	WILL BE READ IN. IF IRUNNC.EQ.0/STOP THE PROGRAM WILL	020377
C	BE STOPPED	020378
C	IDATE = DATE OF RUN, OUTPUTTED FROM SUBROUTINE DATE AS A REAL*8	020379
C	PARAMETER WITH FORMAT (A8)	020380
C	NPAGE = OUTPUT DATA PAGE COUNTER, UPDATED UPON ENTRY INTO PAGEFD,	020381
C	A INTEGER*4 VARIABLE	020382
C		020383
C	THE FOLLOWING PARAMETER IS A REAL*4 ARRAY INPUTTED IN DYN540	020384
C	AND STORED IN /LTITLE/	020385
C	TITLE = TITLE TO BE PRINTED WITH ALL RELEVANT FREQUENCY RESPONSE	020386
C	DATA, READ WITH FORMAT (20A4) DIMENSIONED (20)	020387
C		020388
C		020389
C	DEBUG PRINT CONTROL	020390
C	-----	020391
C	DEBUG = DEBUG PRINT CONTROL TABLE, INPUTTED LOGICAL*4 ARRAY	020392
C	DIMENSIONED (120) AND STORED IN /LDEBUG/	020393
C	IF (IRUNNC.EQ.0/DEBUG) THE DEBUG TABLE IS READ IN USING	020394
C	FORMAT (6CL1) IN SUBROUTINE START	020395
C	IF (DEBUG(K).EQ..TRUE.) EXECISE DEBUG PRINT OPTIONS IN	020396
C	SUBROUTINE DEBUG # K	020397
C	IF (DEBUG(K).EQ..FALSE.) SKIP ALL DEBUG PRINTING IN	020398
C	SUBROUTINE DEBUG # K	020399
C	TO INCLUDE DEBUG PRINT STATEMENTS IN SUBROUTINE DEBUG # M	020400
C	THE FOLLOWING STATEMENTS MUST BE INSERTED	020401
C	DATA NDI/0/	020402
C	LOGICAL DEBUG,LEGU	020403
C	COMMON/LDEBUG/ DEBUG(120)	020404
C	EQUIVALENCE (LEQU,DEBUG(M))	020405

C	TOUCH TO READS PRINT USE	020406
C	IF (CENO) ARIC (NOT,...) ...	020407
C		020408
C		020409
C	WORK AREAS IN MEMORY (DISCOS MAKES USE OF EFFICIENT PROGRAMMING	020410
C	----- PRACTICES BY DEFINING SEVERAL AREAS IN	020411
C	CORE AS WORK AREAS. VARIABLES WHICH MUST	020412
C	BE COMPUTED BUT NEED NOT BE SAVED ARE	020413
C	TEMPORARILY STORED HERE)	020414
C		020415
C	ASUMRY = REAL*8 ARRAY DIMENSIONED (KSUMRY,5) STORED IN /SUMRY/	020416
C	ISUMRY = INTEGER*4 ARRAY DIMENSIONED (KSUMRY,3) STORED IN /SUMRY/	020417
C	FR = TEMPORARY WORK AREA USED IN ADAMS PREDICTOR/CORRECTOR	020418
C	INTEGRATION, DIMENSIONED (KY,5). AREA ALSO USED BY LINEAR	020419
C	IT IS AN ARRAY OF REAL*8 VARIABLES STORED IN /PRWORK/	020420
C	IV = INTEGER*4 WORK AREA USED IN ASIMR AND FINDT FOR STORING	020421
C	INTEGER TABLES USE TO RECORD SIMILARITY TRANSFORMATION	020422
C	MATRIX DERIVED, DIMENSIONED (KY)	020423
C	JV = INTEGER*4 WORK AREA USED IN FINDT IN THE CONSTRUCTION	020424
C	OF IV, DIMENSIONED (KY)	020425
C	W1 = REAL*8 ARRAY DIMENSIONED (L1,L1) USED EXTENSIVELY	020426
C	THROUGHOUT LINEAR STABILITY ANALYSIS FOR THE STORAGE	020427
C	OF LARGE COEFFICIENT MATRICES	020428
C	W2 = USED IN CONJUNCTION WITH W1, REAL*8 ARRAY DIMENSIONED	020429
C	SAME AS W1	020430
C		020431
C		020432
C	ERROR FLAG (SEVERAL ERROR FLAGS ARE DEFINED WITHIN DISCOS	020433
C	----- THEY ARE USED TO INFORM THE USER THAT A PHYSICALLY	020434
C	UNREALIZABLE CONDITION HAS BEEN ENCOUNTERED)	020435
C		020436
C	PHYSICALLY UNREALIZABLE SYSTEM	020437
C	-----	020438
C	NERRR = PROGRAM POINT LOCATOR IN DYN510, AN INPUT ERROR WAS FOUND	020439
C	ERROR MESSAGE FROM DYN510	020440
C	(*INPUT DATA ERROR, NERRR = 0,13)	020441
C		020442
C	ZERO TRANSFER FUNCTION GAIN (ERROR SIGNAL, IF THE DETERMINANT	020443
C	----- OF NUMERATOR ZERO A ZERO GAIN	020444
C	CONDITION ENCOUNTERED)	020445
C	THE FOLLOWING ARE INTEGER*4 VARIABLES STORED IN /ORATIO/. THEY ARE	020446
C	USED TO DEFINE WHERE CALL TO GAUSS1 IS FROM AND IF ZERO GAIN	020447
C	CONDITION ENCOUNTERED	020448
C	IFL1 = 1 IF THE CALL TO GAUSS1 IS FROM NOMS; EQUALS 0 OTHERWISE	020449
C	IFL2 = 1 IF A ZERO NUMERATOR GAIN CONDITION ENCOUNTERED; IF SO,	020450
C	COMPUTATION JUMPS TO THE ANALYSIS OF THE NEXT REQUESTED	020451
C	TRANSFER FUNCTION; EQUALS 0 OTHERWISE	020452
C		020453
C		020454
C		020455
C	RETURN	020456
C	END	020457

SUBROUTINE EIGVEC(IVC, A, B, W, IRCW, XR, XI, VR, VI, ROOTRE,	020458
1 ROOTIE, NE, NMAX, T2, SW1, CCUNTE, ERR)	020459
C	
IMPLICIT REAL*8 (A-H,O-Z)	020460
C SUBROUTINE TO FIND THE EIGENVECTORS OF A NON-SYMMETRIC MATRIX	020461
C BY A MODIFIED WILKINSON'S INVERSE ITERATION METHOD.	020462
C CONTROL IVC CODE IS	020463
C 1 FIND ONLY THE REGULAR EIGENVECTORS (A X = LAMBDA X)	020464
C 2 FIND ONLY THE TRANSPOSED EIGENVECTORS (AT V = LAMBDA V)	020465
C 3 FIND BOTH TYPES OF EIGENVECTORS.	020466
DIMENSION A(NMAX,NMAX), B(NMAX, NMAX), W(NMAX,4), XR(NMAX),	020467
XI(NMAX), VR(NMAX), VI(NMAX), IRCW(NMAX,2)	020468
INTEGER COUNT, CCUNTE, T2	020469
ROOTR = ROOTRE	020470
ROOTI = ROOTIE	020471
N = NE	020472
N1 = N - 1	020473
NPI = N + 1	020474
IVC1 = IVC - 1	020475
IVC2 = IVC1 - 1	020476
COUNT = 1	020477
CLIM = 1.0D-4	020478
IF(ROOTI) 1, 60, 1	020479
C COMPLEX EIGENVALUE.	020480
C	020481
1 TEMP = - ROOTR - ROOTI	020482
ISW = 2	020483
TEMP2 = ROOTR**2 + ROOTI**2	020484
JJ = 300	020485
DO 600 I = 1, N	020486
IF(T2) 600, 603, 600	020487
600 DO 602 J = 1, N	020488
JJ = JJ + 1	020489
IF(JJ - 251) 602, 601, 601	020490
601 JJ = 1	020491
READ (T2) (W(LL,I), LL = 1,250)	020492
602 E(I,J) = A(I,J)*TEMP + W(JJ,I)	020493
GO TO 605	020494
603 DO 604 J = 1, N	020495
604 E(I,J) = A(I,J)*TEMP + B(I,J)	020496
605 E(I,I) = B(I,I) + TEMP2	020497
606 A(I,I) = A(I,I) - ROOTR	020498
IF(T2 .NE. 0) REWIND T2	020499
GO TO 700	020500
607 IF(ICC) 622, 608, 622	020501
C MATRIX SINGULAR.	020502
622 IF(IVC2) 623, 625, 623	020503
623 DO 624 LL = 1, N	020504
W(LL,2) = 0.00	020505
624 XI(LL) = 0.00	020506
IF(IVC1) 625, 614, 625	020507
625 DO 626 LL = 1, N	020508
W(LL,4) = 0.00	020509
626 VI(LL) = 0.00	020510
GO TO 511	020511
C MATRIX NOT SINGULAR.	020512
608 DO 609 LL = 1, N	020513
W(LL,2) = 1.00	020514
W(LL,3) = 1.00	020515
W(LL,4) = 1.00	020516
609 W(LL,1) = 1.00	020517
609 IF(IVC2) 610, 612, 610	020518
	020519

610	DO 611 I = 1, N	020520
	I2 = IRGW(1,2)	020521
	XI(I2) = W(1,1)*ROOT1	020522
	DO 611 J = 1, N	020523
611	XI(I2) = XI(I2) + A(1,J)*W(J,2)	020524
	IF(IVC1) 612, 500, 612	020525
612	DO 613 I = 1, N	020526
	VI(I) = W(1,3)*ROOT1	020527
	DO 613 J = 1, N	020528
613	VI(I) = VI(I) + A(J,1)*W(J,4)	020529
	GO TO 495	020530
615	CERR = 0.00	020531
	IF(IVC2) 616, 619, 616	020532
616	DO 618 I = 1, N	020533
	XR(I) = -W(1,2)	020534
	DO 617 J = 1, N	020535
617	XR(I) = XR(I) + A(1,J)*XI(J)	020536
618	XR(I) = XR(I)/ROOT1	020537
	IF(IVC1) 619, 633, 619	020538
619	DO 621 I = 1, N	020539
	VR(I) = -W(1,4)	020540
	DO 620 J = 1, N	020541
620	VR(I) = VR(I) + A(J,1)*VI(J)	020542
621	VR(I) = VR(I)/ROOT1	020543
C		020544
C	SEARCH VECTORS FOR LARGEST ELEMENT AND NORMALIZE.	020545
C		020546
627	AMAX = 0.00	020547
	DO 629 L = 1, N	020548
	TEMP = VR(L)**2 + VI(L)**2	020549
	IF(TEMP - AMAX) 629, 629, 628	020550
628	AMAX = TEMP	020551
	I2 = L	020552
629	CONTINUE	020553
	C1 = VR(I2)/AMAX	020554
	C2 = -VI(I2)/AMAX	020555
	DO 630 L = 1, N	020556
	TEMP = VI(L)	020557
	VI(L) = VR(L)*C2 + TEMP*C1	020558
630	VR(L) = VR(L)*C1 - TEMP*C2	020559
	IF(CCUNT .EQ. 1) GO TO 632	020560
	DO 631 LL = 1, N	020561
631	CERR = DMAX1(CERR, DABS(VR(LL) - W(LL,3)), DABS(VI(LL) - W(LL,4)))	020562
632	IF(IVC2) 633, 638, 633	020563
633	AMAX = 0.00	020564
	DO 635 L = 1, N	020565
	TEMP = XR(L)**2 + XI(L)**2	020566
	IF(TEMP - AMAX) 635, 635, 634	020567
634	AMAX = TEMP	020568
	I2 = L	020569
635	CONTINUE	020570
	C1 = XR(I2)/AMAX	020571
	C2 = -XI(I2)/AMAX	020572
	DO 636 L = 1, N	020573
	TEMP = XI(L)	020574
	XI(L) = XR(L)*C2 + TEMP*C1	020575
636	XR(L) = XR(L)*C1 - TEMP*C2	020576
	IF(CCUNT .EQ. 1) GO TO 640	020577
	DO 637 LL = 1, N	020578
637	CERR = DMAX1(CERR, DABS(XR(LL) - W(LL,1)), DABS(XI(LL) - W(LL,2)))	020579

C		020580
C	TEST FOR CONVERGENCE	020581
C		020582
638	IF(CCOUNT .EQ. 1) GO TO 646	020583
	IF(CERR .GE. 1.00-4) GO TO 639	020584
	IF(CERR .GE. CLIM) GO TO 648	020585
	CLIM = CERR	020586
	IF(CLIM .LE. 1.00-6) GO TO 648	020587
639	IF(CCOUNT .GE. 40) GO TO 66	020588
647	COUNT = COUNT + 1	020589
	IF(RCUT1) 642, 673, 642	020590
642	IF(IVC2) 640, 644, 640	020591
640	DO 641 LL = 1, N	020592
	X(LL,1) = XI(LL)	020593
641	X(LL,2) = XI(LL)	020594
	IF(IVC1) 644, 610, 644	020595
644	DO 645 LL = 1, N	020596
	X(LL,3) = VI(LL)	020597
645	X(LL,4) = VI(LL)	020598
	GO TO 655	020599
646	CERR = 0.00	020600
	IF(ICC) 646, 647, 648	020601
648	ERR = CERR	020602
	CCOUNT = COUNT	020603
	IF(RCUT1) 667, 668, 667	020604
667	DO 649 I = 1, N	020605
649	A(I,1) = A(I,1) + RCUTR	020606
	RETURN	020607
68	PRINT 101, RCUTR, RCUT1, CERR	020608
	GO TO 648	020609
C		020610
C	REAL EIGENVECTORS.	020611
C		020612
C	HOPE THAT THIS IS THE PROPER PLACE FOR STATEMENT 60	020613
60	CONTINUE	020614
	ISW=1	020615
C	STATEMENT 60 MISSING FROM DATA SUPPLIED	020616
	DO 651 I = 1, N	020617
	DO 650 J = 1, N	020618
650	E(I,J) = A(I,J)	020619
651	E(I,1) = B(I,1) - RCUTR	020620
	GO TO 700	020621
652	IF(ICC) 680, 685, 680	020622
C		020623
C	SINGULAR MATRIX.	020624
C		020625
680	IF(IVC2) 681, 683, 681	020626
681	DO 682 L = 1, N	020627
682	XI(L) = 0.00	020628
	IF(IVC1) 683, 514, 683	020629
683	DO 684 L = 1, N	020630
684	VI(L) = 0.00	020631
	GO TO 511	020632
C		020633
C	MATRIX NOT SINGULAR.	020634
C		020635
685	IF(IVC2) 653, 656, 653	020636
653	DO 654 L = 1, N	020637
654	XI(L) = 1.00	020638
	IF(IVC1) 656, 500, 656	020639

656 DO 657 L = 1, N	020640
657 VI(L) = 1.00	020641
GO TO 455	020642
C	020643
C NORMALIZE REAL VECTORS.	020644
C	020645
655 CERR = 0.00	020646
IF (IVC2) 658, 662, 658	020647
658 C1 = 0.00	020648
C2 = 0.00	020649
DO 660 L = 1, N	020650
TEMP = JAES(XI(L))	020651
IF (TEMP - C1) 660, 660, 659	020652
659 C1 = TEMP	020653
C2 = XI(L)	020654
660 CONTINUE	020655
DO 661 L = 1, N	020656
XI(L) = XI(L)/C2	020657
CERR = DMAX1(CERR, DABS(XI(L) - XR(L)))	020658
661 XR(L) = XI(L)	020659
IF (IVC1) 662, 638, 662	020660
662 C1 = 0.00	020661
C2 = 0.00	020662
DO 664 L = 1, N	020663
TEMP = JAES(VI(L))	020664
IF (TEMP - C1) 664, 664, 663	020665
663 C1 = TEMP	020666
C2 = VI(L)	020667
664 CONTINUE	020668
DO 665 LL = 1, N	020669
VI(LL) = VI(LL)/C2	020670
CERR = DMAX1(CERR, DABS(VI(LL) - W(LL,1)))	020671
VR(LL) = VI(LL)	020672
665 W(LL,1) = VI(LL)	020673
GO TO 638	020674
668 IF (IVC2) 669, 671, 669	020675
669 DO 670 L = 1, N	020676
670 XI(L) = 0.00	020677
IF (IVC1) 671, 70, 671	020678
671 DO 672 L = 1, N	020679
672 VI(L) = 0.00	020680
70 RETURN	020681
C	020682
673 IF (IVC2) 674, 502, 674	020683
674 DO 675 I = 1, N	020684
I2 = IRCW(I, 2)	020685
675 XI(I2) = XR(I)	020686
GO TO 500	020687
499 IF (IVC2) 500, 502, 500	020688
500 DO 501 I = 2, N	020689
I1 = I - 1	020690
DO 501 J = 1, I1	020691
501 XI(I) = XI(I) - B(I, J)*XI(J)	020692
511 IF (IVC1) 502, 514, 502	020693
502 DO 510 I = 1, N	020694
I1 = I - 1	020695
IF (I1) 503, 505, 503	020696
503 DO 504 J = 1, I1	020697
504 VI(I) = VI(I) - B(J, I)*VI(J)	020698
IF (ICC) 505, 506, 505	020699

505 IF(B(I,1)) 506, 507, 506	020700
506 VI(I) = VI(I)/B(I,1)	020701
GO TO 510	020702
507 IF(VI(I)) 508, 509, 508	020703
508 VI(I) = VI(I)*1.00+15	020704
GO TO 510	020705
509 VI(I) = 1.00	020706
510 CONTINUE	020707
IF(IVC2) 514, 525, 514	020708
514 DO 522 I = 1, N	020709
IR = NP1 - I	020710
IF(I - 1) 515, 517, 515	020711
515 I2 = IR + 1	020712
DO 516 J = I2, N	020713
516 XI(IR) = XI(IR) - B(IR,J)*XI(J)	020714
IF(ICC) 517, 518, 517	020715
517 IF(B(IR,IR)) 518, 519, 518	020716
518 XI(IR) = XI(IR)/B(IR,IR)	020717
GO TO 522	020718
519 IF(XI(IR)) 520, 521, 520	020719
520 XI(IR) = XI(IR)*1.00+15	020720
GO TO 522	020721
521 XI(IR) = 1.00	020722
522 CONTINUE	020723
IF(IVC1) 525, 529, 525	020724
525 DO 526 I = 2, N	020725
IR = NP1 - I	020726
I2 = IR + 1	020727
DO 526 J = I2, N	020728
526 VI(IR) = VI(IR) - B(J,IR)*VI(J)	020729
DO 527 L = 1, N	020730
I2 = IRDOW(L,1)	020731
527 VR(I2) = VI(L)	020732
DO 528 L = 1, N	020733
528 VI(L) = VR(L)	020734
529 IF(RCUT1) 615, 655, 615	020735
C	020736
C FACTOR MATRIX.	020737
C	020738
700 ICC = J	020739
SW1 = 1.00+50	020740
DO 701 LL = 1, N	020741
701 IRCW(LL,1) = LL	020742
DO 702 K = 1, N1	020743
AMAX = DABS(B(K,K))	020744
IMAX = K	020745
K1 = K + 1	020746
DO 702 I = K1, N	020747
IF(AMAX .GT. DABS(B(I,K))) GO TO 702	020748
AMAX = DABS(B(I,K))	020749
IMAX = I	020750
702 CONTINUE	020751
IF(AMAX .LT. SW1) SW1 = AMAX	020752
IF(AMAX .GE. 1.00-25) GO TO 723	020753
C(K,K) = 0.00	020754
ICC = ICC + 1	020755
GO TO 702	020756
723 IF(IMAX .EQ. K) GO TO 704	020757
DO 703 J = 1, N	020758
AMAX = B(K,J)	020759

E(K,J) = B(IMAX,J)	020760
703 E(IMAX,J) = AMAX	020761
I2 = IRCW(K,1)	020762
IROW(K,1) = IROW(IMAX,1)	020763
IRCW(IMAX,1) = I2	020764
704 DO 707 I = K1, N	020765
IF(B(I,K)) 705, 707, 705	020766
705 E(I,K) = B(I,K)/B(K,K)	020767
DO 706 J = K1, N	020768
706 E(I,J) = B(I,J) - B(K,J)*B(I,K)	020769
707 CONTINUE	020770
708 CONTINUE	020771
AMAX = CABS(B(N,N))	020772
IF(AMAX - 1.0D-25) 712, 712, 713	020773
712 E(N,N) = 0.0D0	020774
SW1 = 0.0D0	020775
ICC = ICC + 1	020776
GO TO 705	020777
713 IF(AMAX .LT. SW1) SW1 = AMAX	020778
709 IF(ICC .LE. ISW) GO TO 710	020779
PRINT 102, ICC	020780
COUNT = 0	020781
RETURN	020782
710 DO 711 LL = 1, N	020783
I2 = IROW(LL,1)	020784
711 IRCW(I2,2) = LL	020785
IF (ROOT1 .EQ. 0.0D0) GO TO 652	020786
GO TO 607	020787
101 FORMAT(36HMORE THAN 10 LOOPS FOR EIGENVECTOR OF ,2D12.4,	020788
214H DIFFERENCE OF ,D12.4)	020789
102 FORMAT(16H*****WARNING**** , I4, 71H ZEROS ON DIAGONAL OF FACTORED	020790
1 MATRIX. CHECK FOR MULTIPLE EIGENVALUES.)	020791
END	020792

SUBROUTINE ASQR(T2, A, B, NN, NMAX)	020793	
C	DEBUG # 118	020794
IMPLICIT REAL*8 (A-H,O-Z)	020795	
C	020796	
C	PROGRAM TO SQUARE THE A MATRIX. THE MATRIX MUST BE ORIGINALLY	020797
C	IN A. IF T2 IS NON-ZERO, A SQUARED WILL BE STORED ON T2; AND B	020798
C	WILL BE A BUFFER WHICH MUST HAVE AT LEAST 250 WORDS. IF T2 IS	020799
C	ZERO, A SQUARED WILL BE IN B WHICH MUST BE NMAX BY NMAX.	020800
C	020801	
DIMENSION A(NMAX, NMAX), B(NMAX, NMAX)	020802	
INTEGER T2	020803	
N = NN	020804	
11 IF(T2) 12, 13, 12	020805	
12 REWIND T2	020806	
JJ = 0	020807	
DO 3 I=1,N	020808	
DO 3 J=1,N	020809	
JJ = JJ + 1	020810	
E(JJ,1) = 0.0D0	020811	
DO 1 K=1,N	020812	
1 E(JJ,1) = B(JJ,1) + A(1,K)*A(K,J)	020813	

IF (JJ-250) 3,2,2	020814
2 JJ = 0	020815
WRITE (T2) (B(L,1), L = 1,250)	020816
3 CONTINUE	020817
IF (JJ .NE. 0) WRITE (T2) (B(L,1), L = 1,250)	020818
REWIND T2	020819
RETURN	020820
13 DO 14 I = 1, N	020821
DO 14 J = 1, N	020822
B(I,J) = 0.00	020823
DO 14 K = 1, N	020824
14 B(I,J) = B(I,J) + A(I,K)*A(K,J)	020825
RETURN	020826
END	020827


```

IEF1421 - STEP WAS EXECUTED - CCND CCDE 0000
IEF2851 SYS77347.T055525.RV000.NFHPF015.LDDMOD PASSED 0 EXCPS
IEF2851 VCL SER NCS= M2SCR5. DDNAME=PGM=*DD
IEF2851 SYS77347.T055525.RV000.NFHPF015.S0002403 SYSIN 0 EXCPS
IEF2851 VCL SER NCS= M2SCR6. DDNAME=FT05F001
IEF2851 SYS77347.T055525.RV000.NFHPF015.S0002403 DELETED 0 EXCPS
IEF2851 VCL SER NCS= M2SCR0. DDNAME=FT06F001 321 EXCPS
IEF2851 SYS77347.T055525.SV000.NFHPF015.R0002399 SYSOUT
IEF2851 VCL SER NCS= M2SCR2. DDNAME=FT07F001 0 EXCPS
IEF2851 SYS77347.T055525.SV000.NFHPF015.R0002400 DELETED
IEF2851 VCL SER NCS= M2SCR4. DDNAME=SYSPRINT 0 EXCPS
IEF2851 SYS77347.T055525.SV000.NFHPF015.R0002401 DELETED
IEF2851 VCL SER NCS= M2SCR8. DDNAME=SYSUDDUMP 0 EXCPS
IEF2851 SYS77347.T055525.SV000.NFHPF015.R0002402 DELETED
IEF2851 VCL SER NCS= M2SCR2. DDNAME=FT01F001 523 EXCPS
IEF2851 M2.ZNERS.SOURCE
IEF2851 VCL SER NCS= TD6181.
IEF3731 STEP /GO / START 77347.2045
IEF3741 STEP /GO / STOP 77347.2055 CPU IMIN 22.85SEC MAIN 50K LUS OK
- STEP 03 - RETURN CCDE = 0000 STEP TIME = 1.71 MINS=(CPU= 1.38,IO=.33)
IU IN SECS. JISK= 14.80,DRUM= .00,TAPE= 5.39,CELL= .00,OTHR=.24
- SURCHARGES=(DRIVES ALOC=000,TAPE MOUNTS=000,CUKE=000,PAPER=017,PRIORITY=00000)SECS. TOTAL STP TIME= 1.99 MINS. -
IEF2851 SYS77347.T055525.RV000.NFHPF015.LDDMOD DELETED
IEF3751 JOB /NFHPF015/ START 77347.2044
IEF3761 JOB /NFHPF015/ STOP 77347.2055 CPU IMIN 23.49SEC
- SYSTEM=REL21.8E (11-JUL-77) M2
TIME=20.55.32.60 DATE=12-13-77
- JOB 1147- TOTAL TIME = 1.99 MINS=(CPU= 1.39,IO=.60)
IU IN SECS. DISK= 30.86,DRUM= .00,TAPE= 5.39,CELL= .00,OTHR=.54
- SURCHARGES=(DRIVES ALOC=000,TAPE MOUNTS=000,CUKE=000,PAPER=017,PRIORITY=00000)SECS. TOTAL JOB TIME= 2.27 MINS. -
THERE WERE 01 TAPES MOUNTED FOR THIS JOB. TAPE MOUNT CHARGE WAS 00.0 MINUTES.

```

1. Report No. NASA TP-1219	2. Government Accession No.	3. Recipient's Catalog No.	
4. Title and Subtitle A Digital Computer Program for the Dynamic Interaction Simulation of Controls and Structure (DISCOS), Volume II		5. Report Date May 1978	
		6. Performing Organization Code 712	
7. Author(s) Carl S. Bodley, A. Darrell Devers, A. Colton Park, and Harold P. Frisch		8. Performing Organization Report No. G7702-F26	
9. Performing Organization Name and Address Goddard Space Flight Center Greenbelt, Maryland 20771		10. Work Unit No. 506-17-31	
		11. Contract or Grant No.	
		13. Type of Report and Period Covered Technical Paper	
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, D.C. 20546		14. Sponsoring Agency Code	
15. Supplementary Notes			
16. Abstract . This document presents a theoretical development and associated digital computer program system for the dynamic simulation and stability analysis of passive and actively controlled spacecraft. The dynamic system (spacecraft) is modeled as an assembly of rigid and/or flexible bodies not necessarily in a topological tree configuration. The computer program system may be used to investigate total system dynamic characteristics, including interaction effects between rigid and/or flexible bodies, control systems, and a wide range of environmental loadings. In addition, the program system may be used for designing attitude-control systems and for evaluating total dynamic system performance, including time-domain response and frequency-domain stability analyses.			
17. Key Words (Selected by Author(s)) Spacecraft attitude dynamics, Stabilization and control, Flexible body dynamics, Applied mathematics		18. Distribution Statement STAR Category 18 Unclassified-Unlimited	
19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 464	22. Price* \$14.50

*For sale by the National Technical Information Service, Springfield, Virginia 22161.

National Aeronautics and
Space Administration

Washington, D.C.
20546

Official Business

Penalty for Private Use, \$300

SPECIAL FOURTH CLASS MAIL
BOOK

Postage and Fees Paid
National Aeronautics and
Space Administration
NASA-451



NASA

POSTMASTER: If Undeliverable (Section 158
Postal Manual) Do Not Return
